电脑绘图

于　春　解学军　张录州　编著

電子工業出版社
Publishing House of Electronics Industry

内 容 提 要

本书是电脑学习机上机指导系列丛书(共十册)的第八册。重点介绍了键控绘图、F BASIC 绘图、机器语言绘图基础、背景画面的位移、图形字模设计、背景画面的绘制、卡通的控制、背景画面的分裂位移与扭曲等。

书　　名:电脑绘图
著　　者:于春　张学军　张录州
责任编辑:施玉新
特约编辑:董　玲
印 刷 者:北京市顺义李史山胶印厂印刷
出版发行:电子工业出版社出版、发行　URL:http://www.phei.co.cn
北京市海淀区万寿路 173 信箱　邮编 100036　发行部电话:68214070
经　　销:各地新华书店经销
开　　本:787×1092 1/32　印张:7.75　字数:171 千字
版　　次:1997 年 5 月第 1 版　1997 年 5 月第 1 次印刷
印　　数:5000 册
书　　号:ISBN 7-5053-3533-2/TP·1428
定　　价:10.5 元

凡购买电子工业出版社的图书,如有缺页、倒页、脱页者,本社发行部负责调换
版权所有·翻印必究

前　言

家庭电视游戏业的发展，突飞猛进、一日千里。随着游戏机的普及，各种以任天堂系列 8 位游戏机为主机、配备键盘而构成的“普及型家庭电脑学习机”也迅速推出。目前生产厂已有三十余家，产品品种近百个。较有名的品牌有：中山小霸王、兰州金字塔、北京裕兴以及深、广一带的金钥匙、和声、科达、科特、美猴王、宏源、智源等，产品档次也已拉开了低、中、高三个级别。有关资料表明，目前我国电脑学习机的拥有量已逾 800 万台。

现在，学习机的功能日趋完善，实用性越来越强。除小霸王还继续维持其寓教于乐的游戏特点以外，各大生产厂家都向提高学习机的实用性进军。其工作系统已由从日本引进时的单一的 F BASIC（整数）语言发展到中文 BASIC（浮点）语言、LOGO 语言，汇编语言、中文编辑、文字处理等系统。就硬件来说，94 年初裕兴率先在其 YX4 型机上推出标准微机 101 键盘，随后和声、科达、金钥匙等立即推出了仿 101 键盘的产品，小霸王、金字塔也于 10 月份推出了仿 101 键盘的 586 和 9588 产品；金字塔在 94 年率先推出了外接中华机 140K 容量的 5.25 英寸软盘驱动器之后，95 年新春伊始裕兴又推出了与微机完全兼容的 3.5 英寸标准软驱。从应用软件来看，发展更加迅速，已推出了小学、中学乃至高中的各年级辅助教学软件，特别是裕兴推出的仿“金山北大 WPS 6.0”的“裕兴 WPS 文字处理系统”，不仅操作与微机一致，而且文件可与微机相互交流。从而，使学习机的档次又升一级而迈进了电脑阶层，使它变成了名符其实的“家庭电脑”（Family Computer），完全

可以充当现代家庭管理、办公自动化管理的廉价微机。

为了使电脑初学者能够迅速入门，进而登堂入室。我们依据循序渐进的原则，编写了这一套丛书。本套丛书分：

操作人门
文字处理
游戏 BASIC 语言
中文 BASIC 语言
LOGO 语言
汇编语言
电脑作曲
电脑绘图
BASIC 语言程序集
硬件与维修

共十册。

本书是丛书的第八册。分别面向初、中、高三个层次的读者，讨论了键控绘图、F BASIC 语言编程绘图和机器语言编程绘图的方法、步骤。详细地介绍了任天堂游戏背景画面的结构组成特点。系统地论述了背景画面的横向卷动、纵向滚动、画面的分裂位移、画面扭曲等图形特技的控制原理和实现方法。书中还列出了大量的工具程序供编程中直接套用。

本书内容详实、深入浅出，既适合初学者自学，也可作为编写任天堂游戏软件、电脑学习机应用软件的培训教材使用。

本书由苏子栋教授审定，在编写过程中，陈建宇、韩玉文、李观勇、于勇、于腾潇等同志作了大量的工作，在此一并表示感谢。

编　者

丛书学习说明

本丛书是面向电脑初学者的科学普及读物。丛书的编写结构安排，从初学者最易接受的角度开始，顺着初学者的学习兴趣循序渐进、逐步深入，引导初学者在不觉中步入电脑的殿堂。

丛书的前五册为基础篇，重点介绍了电脑基础知识、电脑学习机的选购常识以及键盘、软盘驱动器、打印机等微机外设的基础知识；常用汉字输入法和中文文字处理操作方法；F BASIC、中文 BASIC、LOGO 三种计算机语言的指令及编程方法。

丛书的后五册为提高篇，重点讨论了 6527CPU 工作系统的特点、汇编语言的指令及编程方法；音乐板、PLAY 指令、机器语言三种电脑作曲方法；键控、BASIC 指令、机器语言三种绘图方法；精选了 60 个 F BASIC、中文 BASIC 语言程序；最后讨论了电脑学习机的硬件结构原理和基本维修方法。

整套丛书内容连贯，但又独立成章。十本书读者可依序学习，也可在学习了丛书之一《操作入门》后，根据自己的兴趣和爱好挑选学习。如需要进行文字处理的用户，只要再学习丛书之二就可胜任。爱好音乐的朋友可继续学习丛书之七，就可令电脑演奏音乐。若要进一步提高品位，还须学习丛书之五，使用机器语言编程才能演奏美妙的旋律；电脑绘图爱好者，则要在丛书之三、之八上下功夫。许多朋友在学习了游戏 BASIC 语言后，想自己动手编写任天堂游戏程序及各种实用程序，那么丛书之九提供了各种类型的程序范例供你参考。

本丛书的撰写宗旨就是辅导读者掌握电脑的基础知识，有深造要求的朋友，请继续参阅任天堂游戏理论书籍《FBAISC 语言与编程技巧》、《电脑游戏机硬件与编程特技》、《任天堂游戏编程探密》。

电脑绘图综述

相识“任天堂”，尤如进“天堂”，《未来战士》的广阔旷景、《上尉密令》的重叠楼层、《唐老鸭》的深邃地洞、《激龟忍者传》的奇诡水城……，无不激起人们强烈的探求欲望。特别是《双截龙》中左旋右转的链带、《沙罗曼蛇》中沸腾飞跃的岩浆、《魂斗罗》中鳞波闪耀的瀑布、《忍者蛙》中自转不已的黑塔……，在给人们带来身临其境感受的同时，也令人困惑不已……，不仅相问：如此奇妙的背景特技是怎样实现的呢？

许多朋友在学习了 F BASIC 语言后，无不进行编写诸如下棋、猜谜、冒险、斗智等类游戏的尝试，其中既有调试的烦恼，也有成功的喜悦，个中滋味是不可言谕的。但美滋滋的滋味没多久，就产生了更高的追求：电脑学习机既然嫁接于游戏机，怎样才能绘出任天堂游戏中的背景画面呢？

本书将分三个层次介绍电脑学习机的绘图方法：

第一层面向初学者，介绍使用 BG GRAPHIC（背景绘图）功能键控绘图。这种绘图方法，简单、直观、易学、易会，即使三、四岁的儿童也可操纵自如。当然要绘制复杂的图形也须动一番脑筋。

第二层面向已步入电脑学习机之门的朋友，重点介绍使用 F BASIC 语言编写绘图程序的方法、技巧。并介绍部分绘图工具类的子程序供大家编程中套用。

第三层面向更高追求的读者，将从任天堂游戏的显示原

理、屏幕结构、背景绘图、卡通控制等基础理论出发，重点介绍任天堂游戏中常用的绘图方法、连续画面的卷动和滚动、画面的分裂位移技巧等内容，并结合具体游戏绘图实例，讨论机器语言编程绘图的方法、步骤。

PDG

目　录

第一章　键控绘图 …… (1)

第一节　键控绘图系统简介 …… (1)
第二节　绘图指令 …… (3)
第三节　绘图操作 …… (8)
第四节　键控绘图的使用 …… (11)

第二章　F BASIC 绘图 …… (13)

第一节　从五套环到花边图案 …… (13)
第二节　背景特技处理 …… (21)
第三节　动态背景与卡通的结合技巧 …… (33)
第四节　神奇画笔 …… (36)

第三章　机器语言绘图基础 …… (51)

第一节　6527CPU 的显示屏幕结构 …… (51)
第二节　背景显示画面的结构 …… (52)
第三节　卡通显示画面的结构 …… (60)
第四节　屏幕显示状态的设置 …… (66)
第五节　CPU 与 PPU 之间的通信方法 …… (71)
第六节　屏幕显示窗口 …… (77)
第七节　显示系统的配色 …… (78)
第八节　6527CPU 的中断系统 …… (81)

第四章　背景画面的位移 …… (86)

第一节　画面的简单位移 …… (86)

第二节　画面的连续位移……（90）
第三节　画面的往复位移……（94）
第四节　画面的循环位移……（97）

第五章　图形字模设计……（106）

第一节　字模与图形库……（106）
第二节　图形块设计……（111）
第三节　卡通字模的结构……（113）
第四节　图形的放大……（115）
第五节　自制图形的使用……（120）
第六节　电脑学习机的显示方式……（121）

第六章　背景画面的绘制……（134）

第一节　局部画面的显示……（134）
第二节　直接送显法全屏幕绘图……（144）
第三节　BS·2A 背景绘图法……（155）
第四节　置色绘图法……（168）
第五节　优化查表法绘图……（179）
第六节　间接分段绘图法……（191）

第七章　卡通的控制……（196）

第一节　两个系统中卡通的区别……（196）
第二节　操纵器采样……（199）
第三节　定义卡通……（208）
第四节　《大赛车》卡通定义程序简介……（219）

第八章　背景画面的分裂位移与扭曲……（229）

第一节　背景画面结构的新概念……（229）
第二节　背景画面的分裂位移……（232）
第三节　背景画面的扭曲控制……（239）

第一章　键控绘图

一张白纸，没有负担，好写最奇最妙的文章，好绘最新最美的图画。

F BASIC 有一个易学、易会、易掌握，最简、最便、最直接的绘画方法——BG GRAPHIC（键控绘图）。

BG GRAPHIC（Background Graphic ）是“使用背景图形绘画”的意思。是指利用系统内部储存的具有 4 种颜色变化的 104 种图形进行绘图。

这种绘图方法与任何电脑语言无关，直接使用按键，选择需要的图形，放置在恰当的位置即可。因此，称为“拼图”更确切。它就像搭积木一样，即使三四岁的儿童也能绘出美丽的图案。下面介绍操作方法。

第一节　键控绘图系统简介

进入游戏 BASIC 后，在主菜单下按数字键“2”就进入了 BG GRAPHIC 状态，显示图 1－1 所示的画面。

画面说明如下：

1. 光标

光标为一空心方框“□”，由它指示放置图形的位置。绘图中，使用方向键移动光标到绘图位置。

2. 光标的坐标

当光标在屏幕上移动时，其所处的坐标值能自动跟踪，

并显示。显示范围：

横坐标 X：0～27

纵坐标 Y：0～20

它指明了键控绘图的有效区域为 28 列×21 行。我们知道 F　BASIC 画面的显示分辨率为 28 列×24 行，键控绘图画面保留了下面的三行用于绘图提示。

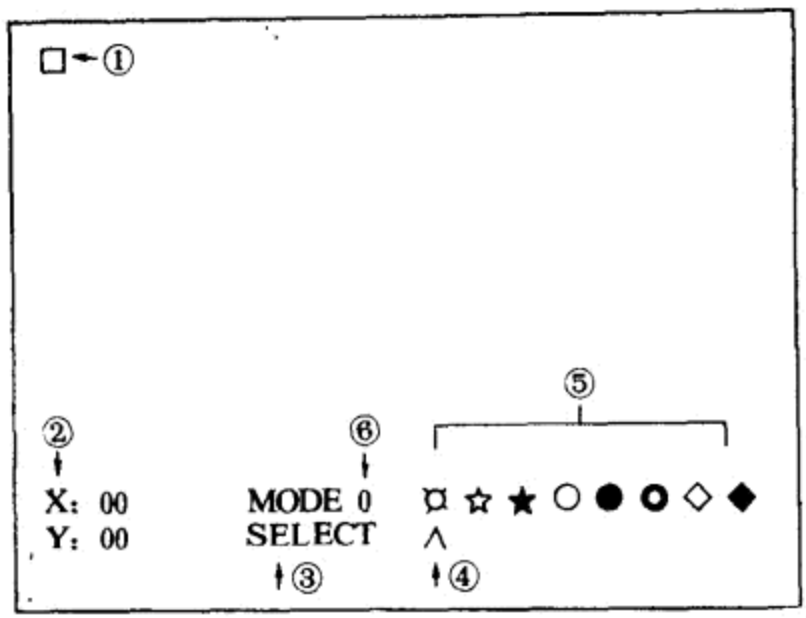


图 1—1　键控绘图画面

3. 操作状态提示

键控绘图共有六条指令，控制进入六个工作状态。系统一旦进入某条指令状态，就在该位置显示某条指令。

4. 图形选择指示符

指示符“ ^ ”用于提示当前绘图所用的图形，可以使用“Ins”、“Del”键控制“ ^ ”向左右移动，选择绘图所用的图形。

5. 图形

键控绘图可以使用的图形共有 104 个，分为 13 组，每组

8个，每次显示一组，图形分组见封底。

图形组的更换方法有两种方式：

对于非101键盘或仿101键盘的学习机，采用“Home”、“Shift＋Home”键更换图形，按一次“Home”键，图形组号加一，即向后翻一组图形；按一次“Shift＋Home”键，图形组号减一，即向前翻一组图形。

对于采用标准101键盘的学习机，则采用“Page Up”、“Page Down”键。“Page Up”控制向前翻一组图形；“Page-Down”控制向后翻一组图形。

6. 配色号码

图形颜色分为4种，颜色编号为0～3。颜色代码的改变可以使用“Enter”键，每按一次，颜色代码加1，在0～3之间循环改变。

第二节　绘图指令

键控绘图系统有六条指令，可通过指令菜单选择。指令菜单使用“Esc”键激活。按“Esc”键后，屏幕左上角将弹出绘图操作指令菜单如图1－2所示。

菜单中，共有6条指令，分别为：“选择背景图形绘画、画面复制、画面移动、画面清除、画面的存储和读出、在画面上显示字符”。

“＞”为指令选择光标，用上、下方向键（↑、↓）可调整“＞”上下移动，选定指令后，按空格键或回车键即可进入相应的指令状态。在任意指令状态下，只要按“Esc”键，就可返回图1－1的画面。

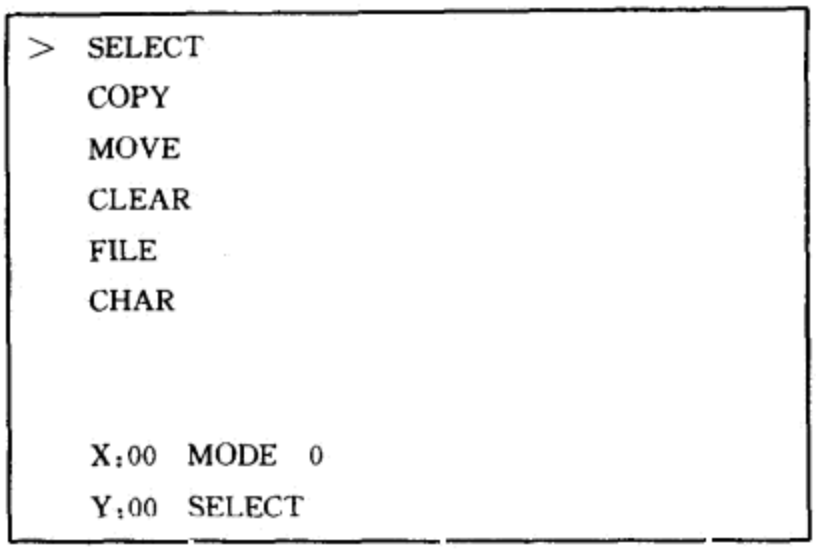


图 1－2　绘图指令选择菜单

一、选择背景图形绘画指令——SELECT

进入键控绘图状态后，系统自动设定为“SELECT”状态。在该状态下，可以自由选择背景图形和设定颜色进行绘画。绘画方法如下：

1. 按“Home”或“Shift＋Home”键，调整屏幕右下角的图形供绘画选择。

2. 按“Del”或“Ins”键，调整图形指示符“^”的位置，选择绘画的图形。

3. 按“Enter”键，设定图形的配色。

4. 使用方向键，调整光标到绘画的位置。

5. 按空格键，所选定的图形便在光标处显示，同时光标后移一格。若要连续显示该图形，可连续按空格键。

6. 如果绘画中出现错误，可把光标移到错误图形的上面，然后按“D”键，即可擦去光标下的图形；也可选择新的图形覆盖旧的图形。

二、复制图形指令——COPY

COPY 指令可在另一位置对画面上已经绘制的图形进行复制，复制后原来的图形依然保留。从而省略了到图形库中选择图形的重复操作，为在一个画面中绘制多个相同的图案提供了方便。

复制操作步骤如下：

1. 按“Esc”键，使用“↓”键从显示的指令群中选择“COPY”指令，然后按空格键，即进入图形复制工作状态。同时，在图 1－1 的操作指令提示一栏将显示“COPY”。

2. 移动光标与复制原图重合。

3. 按一次“Ins”键，确认复制的图形（此时光标下的图形便被记忆在光标中）。

4. 使用方向键移动光标到复制新图的位置上。

5. 按“Del”键，将完成一个图形的复制。

6. 反复使用方向键和“Del ”键可把一个图形在不同的位置上复制多次。

7. 再按“Ins”键，可解除随光标一起移动图形的记忆，并删除记忆的图形。

8. 当欲擦除画面上的某个图形时，可用光标套住该图形，按“Del”键即可擦除。

9. 当复制工作完成，可按“Esc”键，弹出指令菜单，这时指令选择光标自动指向“SELECT”指令，按空格键即退出图形复制状态，返回选择图形绘制状态。

三、图形位置移动指令——MOVE

MOVE 指令可将画面上的图形移到另外的地方，同时

原位置上的图形消失。

操作步骤如下：

1. 按“Esc”键，调出指令选择画面，移动“>”符号到“MOVE”指令行，按空格键，进入图形移动位置工作状态。

2. 使用方向键，移动光标到欲移动图形的位置上。

3. 按“Ins”键确认移动的图形。

4. 使用方向键，使光标和确认的图形移到预定的位置上，同时原图形消失。

5. 按“Del”键，把光标中的图形放置在光标当前的位置上。

6. 图形的删除：

在 MOVE 指令状态下删除图形，可先确定光标的位置，然后再按两次“Ins”键，光标下的图形即被删除。

7. 退出图形移动工作状态的方法与图形复制相同。

四、清屏指令——CLEAR

当欲清除屏幕上的所有图形时，可选择清屏指令 CLEAR。

操作步骤如下：

按“Esc”键进入指令选择画面，移动“>”符号指向“CLEAR”，按空格键即可完成清屏。

五、显示字符指令——CHAR

在 CHAR 状态下，可使用键盘把字母、数字及符号显示在画面的任意位置。

操作步骤如下：

1. 按“Esc”键进入指令选择画面，选择 CHAR 指令，按

空格键进入 CHAR 工作状态。

2. 移动光标到显示字符的位置。

3. 直接敲键盘，即可把键盘上的字符显示在光标位置（注——显示的字符将覆盖光标下的图形）。

4. 退出字符输入工作状态的方法与图形复制相同。

六、画面的存储和调出指令——FILE

在 FILE 工作状态下，可将绘制的图形存入录音机磁带中保存，也可从磁带中调出保存的图形。对配置磁盘驱动器的学习机，也可以在该状态下把图形存入磁盘或从磁盘中调出图形。

操作步骤如下：

1. 按“Esc”键进入指令选择画面，选择“FILE”指令，按空格键将进入 FILE 工作状态。

2. 屏幕清屏，显示

SAVE(S),LOAD(L)?

询问是保存(SAVE)文件还是调出(LOAD)文件，若保存文件则按“S”键，若调出文件则按“L”键。

3. 按“S”键(或“L”)键后，将在第二行显示

FILE NAME"

要求输入保存(或调出)的文件名。

4. 录音机连接好录音线(若保存文件，录音线的一端插入学习机的“SAVE”插口，另一端插入录音机的“MIC”插口；若调出文件，录音线的一端插入学习机的“LOAD”插口，另一端插入录音机的耳机插口。注——录音机的调整见丛书之三《中文 BASIC》)，输入文件名如“PICTURE”。

5. 若是保存文件，首先把空白磁带放入磁带仓，把磁带

倒到有记号的位置，同时按下录音机的放音、录音键；然后按“Enter”键，将在屏幕的第三行显示

WRITING PICTURE

这时，屏幕锁定不动，稍停一会文件保存完毕，自动返回“SELECT”状态画面。

6. 若是调出文件，首先把装有调入文件的磁带放入带仓，并把磁带倒在保存文件的开始处，按“Enter”键；然后按下录音机的放音键。这时，屏幕锁定。若读到欲调入的文件将在第三行显示

LOADING PICTURE

调入完毕将自动返回“SELECT”工作状态，同时调入的绘画也显示在屏幕上。

7. 若是对磁盘操作，则在输入文件名前，首先输入“A:”、再输入文件名，即可把图形存入磁盘或从磁盘中调出图形。

第三节　绘图操作

一、机器人绘制步骤

本节以绘制如图 1—3 所示的机器人为例，介绍具体的键控绘图方法。

绘画步骤如下：

从第 4 组图形中选择第 5 号图形画眼睛。

从第 6 组图形中选择第 5、6、7 号图形画脸。

从第 4 组图形中选择第 2 号图形画左边的脖子。

从第 1 组图形中选择第 2 号图形画右边的脖子。

从第 2 组图形中选择第 7、8 号图形画肩、胸。

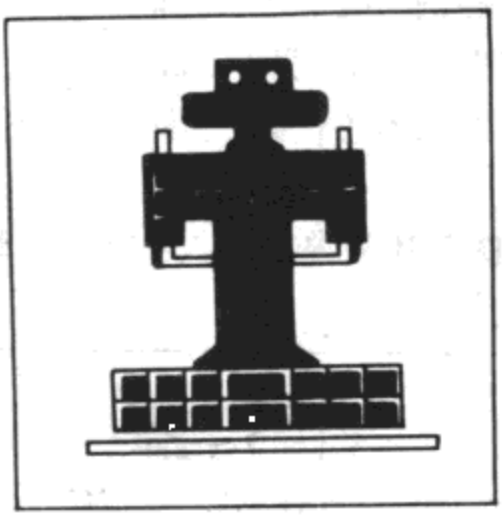


图 1—3　机器人

从第 1 组图形中选择第 5 号图形画左边的手臂。

从第 4 组图形中选择第 5 号图形画右边的手臂。

从第 3 组图形中选择第 5 号图形、第 2 组选第 1 号图形画上腹部。

从第 1 组图形中选择第 1、2 号图形画下腹部。

从第 1 组图形中选择第 3 号图形画左腿。

从第 3 组图形中选择第 7 号图形画右腿。

从第 2 组图形中选择第 2 号图形画左脚。

从第 3 组图形中选择第 2 号图形画右脚。

从第 3 组图形中选择第 7 号图形画左边台阶。

从第 4 组图形中选择第 8 号图形画右边台阶。

从第 10 组图形中选择第 1、4、2、5、7、8 号图形画连接线及底线。

至此，逗人的机器人呈现在你的面前。

二、绘图叙述规则

读者可能觉得上述绘图过程的叙述较繁琐，每个图形的定位不明确，开始绘图时要试多次才能安放恰当。因此，绘图叙述中还应加入图形坐标。另外，每个图形还有 4 种颜色变化，故叙述中还应加入颜色代码。这样一来，绘图的叙述将变得很繁琐。

为使键控绘图叙述简单明了，今后将采取图 1—4 所示的形式进行说明。

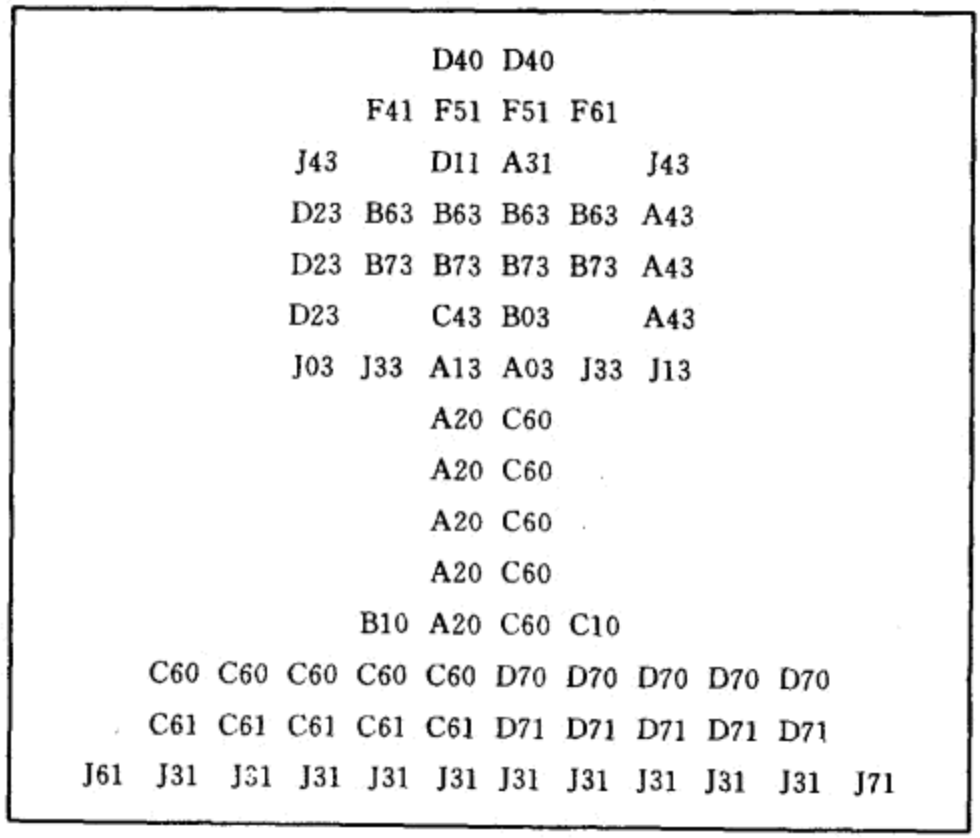


图 1—4　键控绘图说明方法

图中每一组字符串表示一个固定颜色的图形。如："B73"表示背景图形库中(封底)第"B"行第 7 个图形，其颜色代码为"3"。每个图形的纵、横坐标可从图中直接查出，对照光标的提示坐标即可对图形进行准确地定位。

第四节　键控绘图的使用

当欲结束键控绘图工作状态时，可先按“Esc”键，再按“Pause”（或“Stop”）键，即可返回 BASIC 主菜单。

在 BG GRAPHIC 状态下绘制的图形，可在 BASIC 程序的运行中调出使用。调出方法是：在 BASIC 程序中，使用“VIEW”指令代替清屏指令“CLS”。运行程序后，在清屏的同时，将显示键控绘图的画面。如绘制了图 1－5 中的“山和星星”画面或如图 1－6 中“树和岛”的画面后。返回 BASIC 编程状态，直接执行“VIEW”命令，将调出图形；若在编写的游戏程序中，加入“VIEW”语句行，将在执行该行语句时，调出图形。

键控绘制的图形保存在显示映射区的第二页，调出和清屏指令对它都没有影响，因此可以反复调用。

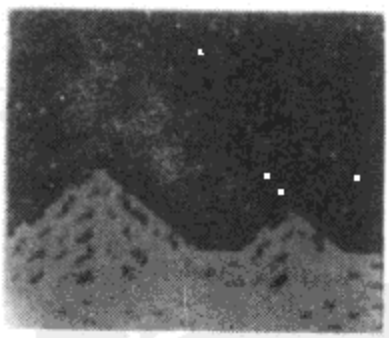


图 1－5　山和星星

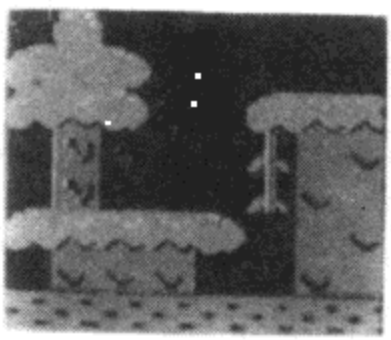


图 1－6　树和岛

键控绘图的最大特点是：与 BASIC 工作方式互相独立，图形数据不占用用户 RAM 区。因此，采用键控绘图方法绘制画面编写应用程序或游戏程序，可以压缩程序量，减少图

形对 RAM 区的占用(丛书之九中的许多游戏程序就是使用键控绘图方法编写的),使程序短小精悍、结构紧凑,而且画面效果直观,程序便于阅读和交流。

第二章　F BASIC 绘图

键控绘图虽然具有效果直观、操作简单、修改方便等优点，但在程序运行中，画面的内容不能改变，因此只适用于画面固定的场合。对于程序运行中，要经常改变画面内容或画面内容随机出现的场合，如多关次、多画面的游戏，键控绘图就显得捉襟见肘、力不从心了。

F BASIC 语言编程绘图，具有动态显示、画面随机、绘图灵活的优点。但是对编程的技能要求较高，编程的技巧性较强。使用较小的程序绘制变化繁复的画面，是编程功底深厚的具体体现。

本章将从简单图形的绘制和变化入手，介绍编程绘图的一般方法技巧和背景图形的特技处理，并给出一个使用操纵器控制绘图的工具程序《神奇画笔》，最后以《太空探险》游戏程序为例，介绍背景画面与卡通图形的结合编程方法。

第一节　从五套环到花边图案

一、绘制五套环

例 2—1　编程打印如图 2—1 所示的五套环图形。

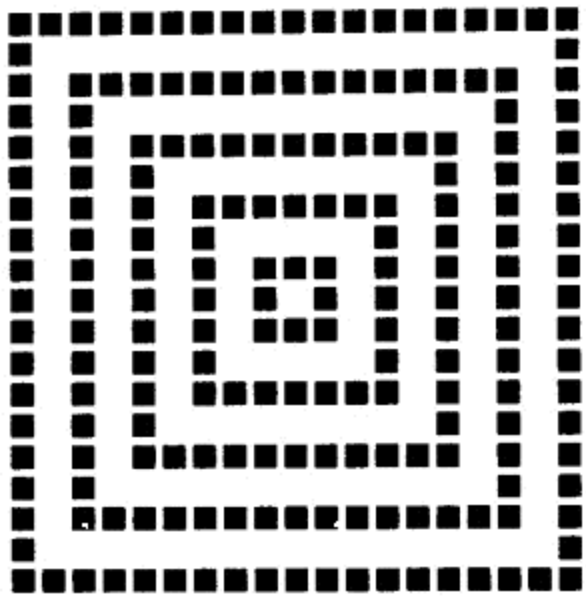


图 2－1　五套环

程序见 No. 2－1。

```
10 REM Nc. 2－1, WU TAO HUAN 1
20 CLS: X＝13: Y＝11
30 A $ ＝CHR $ (RND(70)＋185)
40 FOR N＝1 TO 9 STEP 2
50 FOR L＝－N TO N
60 LOCATE X＋L,Y－N: PRINT A $
70 LOCATE X＋N,Y＋L: PRINT A $
80 LOCATE X－L,Y＋N: PRINT A $
90 LOCATE X－N,Y－L: PRINT A $
100 NEXT: NEXT
110 LOCATE 0,22: END
```

运行程序，将顺时针打印图 2－1 所示的图形。

二、反复绘制、擦除五套环

例 2－2　编程打印如图 2－1 所示的图形。并使五套环按顺时针方向从内向外绘制，绘完后，再令其按逆时针方向从外向内消失。消失后再绘制，周而复始。

程序见 No. 2－2。

```
10 REM No. 2－2, WU TAO HUAN 2
20 CLS: X=13: Y=11: A$=CHR$(253): B$="  "
30 A$=CHR$(RND(70)+185)
40 FOR N=1 TO 9 STEP 2
50 FOR L=－N TO N
60 GOSUB 200: NEXT: NEXT
70 SWAP A$,B$
80 FOR N=9 TO 1 STEP －2
90 FOR L=N TO －N STEP －1
100 GOSUB 200: NEXT: NEXT
110 SWAP A$,B$: GOTO 30
200 LOCATE X+L,Y－N: PRINT A$
210 LOCATE X+N,Y+L: PRINT A$
220 LOCATE X－L,Y+N: PRINT A$
230 LOCATE X－N,Y－L: PRINT A$
240 RETURN
```

程序说明：

30 行用于随机产生每次绘图的图形。

40～60 行为绘图程序。

80～100 行用于擦除图形。

70、110 行用于交换两字符变量中的内容。

200～240 行为绘图子程序。

若使擦图也按顺时针方向从外向内进行，可把 90 行改为

90 FOR L=－N TO N

则可令绘图、擦图全部以顺时针方向进行。

三、五套环程序的优化

若用绘制新图的方法擦除旧图，则可删除 70～100 行。那样程序将更简洁。见程序 No. 2－3。

```
10 REM No. 2－3, WU TAO HUAN 3
20 CLS: X=13: Y=11
30 A$=CHR$(RND(70)+185)
40 FOR N=1 TO 9 STEP 2
50 FOR L=－N TO N
60 LOCATE X+L,Y－N: PRINT A$
70 LOCATE X+N,Y+L: PRINT A$
80 LOCATE X－L,Y+N: PRINT A$
90 LOCATE X－N,Y－L: PRINT A$
100 NEXT: NEXT
110 SWAP A$,B$: GOTO 30
```

运行程序，五套环将以各种图案绘制，周而复始……。

四、五套环的变形

1. 花窗图案

若把程序 No. 2－3 的 50 行改为

50 FOR L=－N－2 TO N+2

将以各种图形反复绘制如图 2-2 所示的花窗图案。

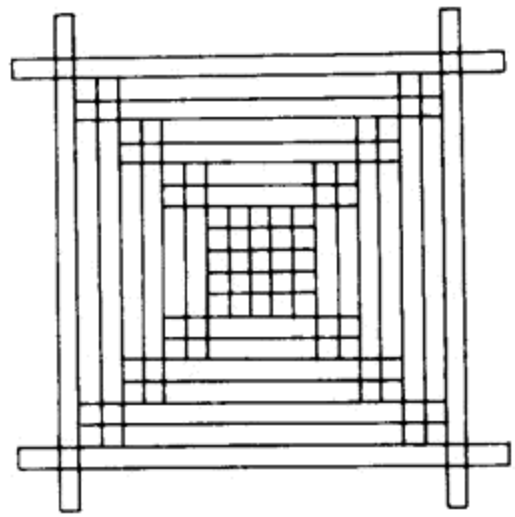


图 2－2　花窗图案(一)

2. 五风车嵌套图案

若把程序 No. 2－3 的 50 行改为

50 FOR L＝－N＋1 TO N＋2

将按以上的方式绘出图 2－3 所示的五风车嵌套图案。

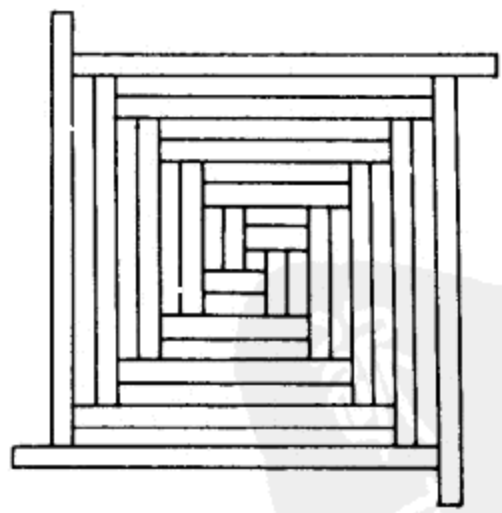


图 2－3　五风车嵌套图

3. 更精美的花窗

若把程序 No. 2－3 的 40、50 行改为

40 FOR N＝1 TO 7 STEP 2

50 FOR L＝－N－3 TO N＋3

将绘制出更精美的花窗图案(见图 2－4)。

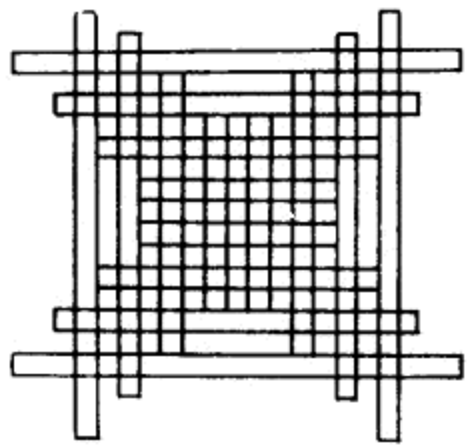


图 2－4　花窗图案(二)

4. 万变图

若把程序 No. 2－3 改为 No. 2－4 的形式，绘制出的图案将出你意料之外。

```
10 REM No. 2－4, WAN BIAN TU
20 CLS: X＝13: Y＝11
30 A$ ＝CHR$ (RND(70)＋185)
40 FOR N＝1 TO 7 STEP 2
50 FOR L＝－RND(N＋3) TO RND(N＋3)
60 LOCATE X＋L,Y－N: PRINT A$
70 LOCATE X＋N,Y＋L: PRINT A$
80 LOCATE X－L,Y＋N: PRINT A$
90 LOCATE X－N,Y－L: PRINT A$
```

100 NEXT：NEXT

110 PAUSE：CLS：GOTO 30

运行程序，将打印出一种图案。按任意键后又打印一种图案。图案千变万化，美不胜收。现仅录其三幅于图 2—5～2—7 供欣赏。

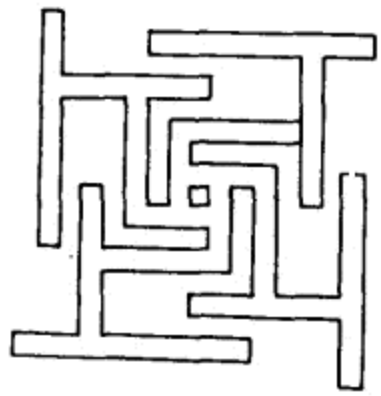


图 2—5　万变图(一)

图 2—6　万变图(二)

连续绘制 1000 幅，你能找出两幅相同的图案吗？

5. 花式墙体构图

若把程序 No. 2—3 改为 No. 2—5 的形式，将设计出各种墙体结构图样。假如你是一位建筑设计师，不知将有如何感想。

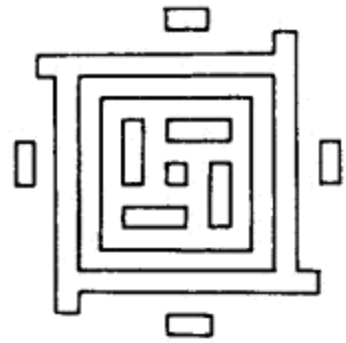


图 2－7　万变图(三)

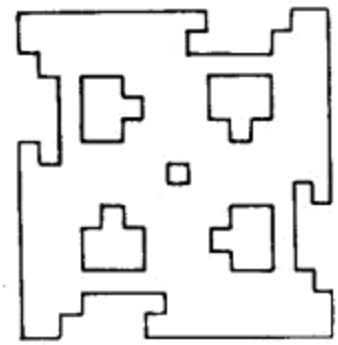


图 2－8　墙体结构图(一)

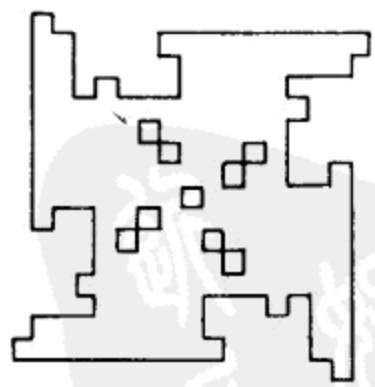


图 2－9　墙体结构图(二)

```
10 REM No. 2－5, QIANG TI GUO TU
20
```

```
20 CLS: X=13: Y=11
30 A$=CHR$(253)
40 FOR N=1 TO 7 STEP RND(2)+1
50 FOR L=-RND(N+2) TO RND(N+2)
60 LOCATE X+L,Y-N: PRINT A$
70 LOCATE X+N,Y+L: PRINT A$
80 LOCATE X-L,Y+N: PRINT A$
90 LOCATE X-N,Y-L: PRINT A$
100 NEXT: NEXT
110 PAUSE: CLS: GOTO 40
```

仅录两幅供欣赏，见图 2－8、2－9。

第二节　背景特技处理

特殊的背景显示，如标题字符从屏幕的一边缓慢移入、动态清除屏幕、显示字符时配合着滴滴的音响等效果，都将为应用程序或游戏程序增辉添彩。本节将介绍常用背景特殊处理的一般技巧。

一、字符的动态显示

若在游戏开始时，标题字符或有关说明从屏幕的一侧缓慢移出，并周尔复始的连续运动，将使呆板的标题介绍给人以清新活泼的感受。程序 No. 2－6 可达到这一目的。

```
5  REM No. 2-6 CAPTIONS MOVE
10  CLS: CLEAR: N$="* * * TAI KONG TAN XIAN
    * * *": M$=N$:
    Z$="WELCOME TO PLAY!"
```

```
20  N$=MID$(N$,2,24)+LEFT$(N$,1)
30  Z$=MID$(Z$,2,15)+LEFT$(Z$,1)
40  M$=RIGHT$(M$,1)+MID$(M$,1,24)
50  LOCATE 1,0: PRINT N$
60  LOCATE 6,10: PRINT Z$
70  LOCATE 1,20: PRINT M$
80  PAUSE 20: GOTO 20
```

运行程序，上一行将循环左移显示“太空探险”的游戏标题；下一行循环右移显示“太空探险”；中间一行循环左移显示“欢迎游戏”。这就像把字符印在电度表的铝盘上面，通电后，铝盘旋转，字符循环在窗口显示。

改变 80 行中 PAUSE 语句的参数，可以调整字符移动的速度。如在程序中加入读操纵器的语句

```
75  K=STRIG(0): IF K=1 GOTO 100
100 （游戏的开始处理程序入口）
```

则可使用操纵器的“START”键控制游戏的开始。

二、动态清屏

直接使用“CLS”指令清屏，清屏过程过于单调，下面给出 6 个动态清屏程序 No. 2－7～No. 2－12，可使简单的清屏操作变为富有情趣的动画过程。

1. 由屏幕的上、下边开始向中间横向清屏

程序见 No. 2－7。

```
5  REM No. 2－7（上、下向中间清屏）
10  FOR I=1 TO 21: PRINT "AAAAAAAAAAAAA
    AAAAAAAAAAAAAA":NEXT
15  B$="                    "(28 个空格)
```

```
20  FOR I=1 TO 12
25  LOCATE 0,I: PRINT B$
30  LOCATE 0,23-I: PRINT B$
35  NEXT: LOCATE 0,0
```

2. 由屏幕的中间开始向上、下边横向清屏

程序见 No. 2－8。

```
5   REM No. 2-8 (中间向上、下边清屏)
10  FOR I=1 TO 21: PRINT "AAAAAAAAAAAAA
    AAAAAAAAAAAAAA": NEXT
15  B$ ="            "(28 个空格)
20  FOR I=1 TO 12
25  LOCATE 0,12-I: PRINT B$
30  LOCATE 0,11+I: PRINT B$
35  NEXT: LOCATE 0,0
```

3. 由屏幕的中间开始向左、右边纵向清屏

这种清屏效果犹如拉开帷幕。程序见 No. 2－9。

```
5   REM No. 2-9 (中间向左、右边清屏)
10  FOR I=1 TO 672: PRINT "C";: NEXT
20  I=1
30  FOR J=0 TO 22
40  LOCATE 14-I,J: PRINT "  "
50  LOCATE 13+I,J: PRINT "  "
60  NEXT
70  I=I+1: IF I=15 THEN 90
80  GOTO 30
90  LOCATE 0,0: END
```

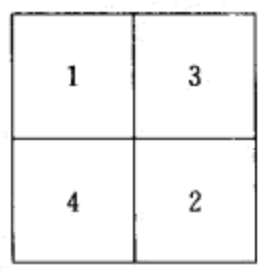
4. 从屏幕的左、右边开始向中间纵向清屏

这种清屏效果犹如合上帷幕。程序见 No. 2－10。

```
5  REM No. 2－10（两边向中间清屏）
10  FOR I=1 TO 672: PRINT "D";: NEXT
20  I=0
30  FOR J=0 TO 22
40  LOCATE I,J: PRINT "  "
50  LOCATE 27－I,J: PRINT "  "
60  NEXT
70  I=I+1: IF I=15 THEN 90
80  GOTO 30
90  LOCATE 0,0: END
```

5. 把屏幕分成四部分交叉清屏

这种清屏操作是把屏幕分成如下图所示的四部分，再交叉顺序清屏。清屏顺序为：



程序见 No. 2－11

```
5  REM No. 2－11（四部分交叉清屏）
10  FOR I=1 TO 672: PRINT "E";: NEXT: T=0: U=
    0: V=0
20  FOR Y=0 TO 11
30  LOCATE U,Y+V: PRINT "  "（14 个空格）
40  NEXT: T=T+1
```

```
50  IF T=1 THEN U=14: V=11: GOTO 20
60  IF T=2 THEN V=0: GOTO 20
70  IF T=3 THEN U=0: V=11: GOTO 20
80  END
```

6. 从外向内右旋转清屏

程序见 No. 2－12。

```
5   REM No. 2－12 (外向内右旋转清屏)
10  FOR I=1 TO 21: PRINT "FFFFFFFFFFFFFFF
    FFFFFFFFFFFFF": U=0
20  FOR I=U TO 26－U: LOCATE I,U: PRINT " ":
    NEXT
30  FOR J=U TO 21－U: LOCATE 27－U,J: PRINT "
    ": NEXT
40  FOR K=26－U TO U+1 STEP －1: LOCATE K,21
    －U: PRINT " ": NEXT
50  FOR L=21－U TO U+1 STEP －1: LOCATE U,L:
    PRINT " ": NEXT
60  U=U+1: IF U=11 THEN LOCATE 0,0: END
70  GOTO 20
```

以上程序结构较简单，不多说明。读者只要掌握了清屏程序的设计思路，定能编写出许多生动有趣的清屏程序。

三、动态背景显示

许多任天堂游戏节目，如《火箭车》、《机车大赛》、《南极探险》等，都采用了移动背景画面的手法以产生卡通移动的效果(实际上卡通的 Y 坐标是固定的)。使用 F BASIC 语言

也可以模仿出这种动态背景效果。

设想一艘太空船要经过太空通道降落到某个星球上。我们可以设定太空船的Y坐标不变，令太空通道和太空陨石从太空船两侧飞速上移，就会产生太空船飞速下降的效果。

程序清单见No. 2－13。

```
5  REM No. 2－13 BG MOVE
10  CLS: SPRITE ON: CGSET 0,1
15  DIM A$(14)
20  FOR I=0 TO 14
25  A$(I)=CHR$(207+I)
30  NEXT
35  DEF MOVE(0)=SPRITE(7,0,1,4,0,0)
40  POSITION 0,80,40
45  PRINT A$(RND(15)): LOCATE 17,23
50  MOVE 0: PRINT A$(RND(15))
55  GOTO 45
```

程序中20～30行把打印太空通道的图形装入A$数组；45～55行控制打印左、右太空通道的边界，同时控制太空船运动。

四、背景的显示、消失与重显

在《淘金者》游戏中，小熊用手一指地面，即可在地面上挖出一个小坑，埋掉追赶它的敌人。这就是使用了背景画面的消失与重显技术。实现的方法是：在欲消失背景的位置上打印空格(即用空格覆盖背景)，打印空格前要先用串变量记忆被覆盖的背景代码，恢复显示时，再把记忆的背景重新打印出来。实现方法及示范程序见No. 2－14。

```
5  REM No. 2-14 BG ELIMINATE and RESHOW
10  CLS: SPRITE ON
15  FOR I=0 TO 26 STEP 2
20  X=I: Y=15: GOSUB 65
25  X=I: Y=17: GOSUB 65
30  NEXT
35  DEF MOVE(7)=SPRITE(0,3,1,1,0,0): POSITION
      7, 124,128: MOVE 7
40  S=0: S=STRIG(0)
45  X=(XPOS(7)-12)/8: Y=(YPOS(7)-16)/8
50  IF S=4 THEN X=X-2: GOSUB 80
55  IF S=8 THEN X=X+2: GOSUB 80
60  GOTO 40
65  LOCATE X+1,Y: PRINT CHR$(247): LOCATE
    X,Y:PRINT CHR$(243)
70  LOCATE X,Y+1: PRINT CHR$(245): LOCATE X
    +1,Y+1:PRINT CHR$(248)
75  RETURN
80  A$=SCR$(X,Y+1): B$=SCR$(X+1,Y+1)
85  C$=SCR$(X,Y+2): D$=SCR$(X+1,Y+2)
90  LOCATE X,Y+1: PRINT "  "
95  LOCATE X,Y+2: PRINT "  "
100  PAUSE 100
105  LOCATE X,Y+1: PRINT A$;B$
110  LOCATE X,Y+2: PRINT C$;D$
115  RETURN
```

程序说明：

10～35 行为初始化程序，显示地面和玛丽。

40～60 行为主控程序。

65～75 行为画地面子程序。

80～115 行为地面消失和重显子程序。其中由 80、85 两行记录消失前的背景画面。

运行程序，使用 1# 操纵器的 A、B 键，可控制玛丽在左、右边挖坑。挖坑后延时一段时间自动填平。

若把上述程序使用在游戏程序中，可把 100 行改为 RETURN(返回调用程序)，在主程序中加入一段延时控制语句，如

D=D+1：IF D>100 THEN GOSUB 105

GOTO 程序的循环体入口

等。

五、用操纵器控制闪烁光标移动

在中华学习机的《挖金砖》游戏中，游戏者可使用键盘控制闪烁的光标移动，以选择挖取的金砖。电脑学习机中，只有操纵器控制卡通运动的示例，而没有介绍如何使用操纵器控制光标移动。但这种控制形式在游戏程序中经常用到。如应用程序中的菜单选项、游戏程序中的密码输入等，都要用操纵器控制光标移动。

现在，我们来讨论操纵器控制光标移动的方法。

1. 闪烁光标的实现

中华学习机中的闪烁光标是通过对某一固定点交替染黑、白两色实现的。在电脑学习机中，我们也可以采用这一方法。首先使用 PALETB 语句定义黑、白色块代码；然后在显示光标的位置打印某种背景图案，如 M7 等；最后使用 COL-

OR 语句进行染色，从而达到显示闪烁光标的目的。但是，在使用中发现这种方法获得的闪烁光标不理想，它不仅显示的光标块太大(有 4 个字符大小)，而且还要占用两个色块代码，使用也不方便。

获得闪烁光标的另一方法是：在闪烁光标的坐标处直接打印 M5 号背景图案(若一块不够长，可使用两块)；延时一段时间后，打印空格进行覆盖；再延时一段时间后再打印 M5 号图案。这样构成的闪烁光标，大小可随意调节，式样可任意选择，使用灵活方便。

2. 闪烁光标的控制

闪烁光标确定后，可通过读操纵器方向键值来改变光标显示的坐标，从而达到控制光标移动的目的。示范程序见 No. 2－15。

3. 闪烁光标示范程序清单

```
5  REM No. 2－15  CURSOR MOVE MENT
10  CLS: X＝10: Y＝10: A $ ＝CHR $ (253)
20  K＝0: SX＝0: SY＝0
30  K＝STICH(0): IF K＝0 THEN 80
40  IF K＝1 THEN SX＝2
50  IF K＝2 THEN SX＝－2
60  IF K＝4 THEN SY＝1
70  IF K＝8 THEN SX＝－1
80  X＝X＋SX: IF X＞27 THEN X＝26
90  IF X＜0 THEN X＝0
100  Y＝Y＋SY: IF Y＞20 THEN Y＝20
110  IF Y＜6 THEN Y＝6
120  GOSUB 130 : GOTO 20
```

```
130  LOCATE X,Y: PRINT A$;A$: PAUSE 30
140  LOCATE X,Y: PRINT "  ": PAUSE 15
150  RETURN
```

4. 程序说明

20～120 行为光标移动控制处理程序。

130～150 行为光标闪烁控制子程序。

程序中通过 80～110 行限制光标的移动范围在(0, 6)→(0,20)→(26,6)→(26,20)之间。若欲扩大、缩小移动范围，可调整这四行语句中 X、Y 的赋值。

六、背景与音响结合特技

《人间兵器》、《恶魔城传说》等游戏中，在显示游戏说明或字符提示时，随着滴滴的音响，把字符逐个打印在屏幕上，使枯燥的字符显示变得生动活泼。这是使用了显示与音响结合技术实现的，下面介绍实现这一合成的方法。

例如：我们欲在屏幕上开辟的横坐标为 6～22、纵坐标为 4～15 的窗口中，显示如下一段文字时：

Wind and rain escorted Spring's departure,
Flying snow welcomes Spring's return.
On the ice—clad rock rising high and sheer
A flower blooms sweet and fair.

(毛泽东诗词：风雨送春归，飞雪迎春到，已是悬崖百丈冰，犹有花枝俏)

实现程序见 No. 2—16。

```
5  REM No. 2—16
10  CLS
20  PLAY "T1M1V5Y2"
```

```
30  DIM A$(10)
40  FOR I=0 TO 10
50  IF I MOD 3=0 THEN A=A+1
60  LOCATE 6,4+I+A: READ A$(I)
70  FOR J=0 TO LEN (A$(I))
80  C$=MID$(A$(I),J+1,1)
90  PRINT C$;: PLAY "O3G2"
100  NEXT
110  PAUSE 50 : NEXT
120  PLAY "O3E2DCDE": LOCATE 0,22: END
130  DATA "Wind and rain es", "corted Spring's"," de-
     parture, "," Flying snow welc","omes Spring's"," re-
     turn."
140  DATA "On the ice-clad"," rock rising high", " and
     sheer","A flower blooms"," sweet and fair."
```

程序中 50 行的作用是，在字符打印时，每打印完一句，空出一行，以方便阅读。

七、背景图形与程序的合成

某些用 F BASIC 语言编写的游戏软件，往往把背景图案直接写在语句中，这样做的好处是：

- 表述直观、便于阅读理解。
- 可使绘图处理程序的结构简化，程序量减少。
- 可提高背景图形的显示、打印速度。
- 便于背景图形整块移动。
- 在多画面的游戏中，预先把每一幅画面的图形装入一个数组中。当更换画面时，只要通过循环语句打印出数组中

的图形即可达到迅速换场的目的。

但是，大家都清楚，背景图形只能使用

PRINT CHR $ (XXX)

语句打印在屏幕上，怎么把它写入程序中呢？这涉及到程序输入技巧问题。具体实现方法如下：

1. 键控绘图

在 BASIC 工作状态下，键入“SYSTEM”命令进入 BG GRAPHIC 状态，进行键控绘图。

2. 调出画面

绘图完毕，返回 F　BASIC 工作状态，键入“VIEW”命令调出绘制的画面。

3. 装入数组

把画面中的每一行图形作为一个字符串，装入字符串数组中（装入时，先把光标移到行首，按“Ins”键在行首插入一定数量的空格，用于输入行号和串变量名、等号、引号等）。一般每一幅画面用一个串数组，如 A $ (20)、B $ (20)。

4. 使用画面

把 0～20 行图形全部写入程序后，当需要某幅画面时，只要使用

FOR I=0 TO 20 : PRINT A $ (I): NEXT

就可立即调出画面。

5. 注意事项

(1)当用“VIEW”命令调出图形时，可能因出现指令执行完毕提示符“OK”破坏画面的情况，这时可使用

LOCATE X,Y : VIEW

形式的命令，把光标和“OK”移到其它位置显示。

(2)每行图形写入变量时的开始位置要固定，否则调出时

将破坏画面的完整、统一。

(3)使用上述方法绘图，不能使用图形库中前四组图形，否则将不能显示。

第三节　动态背景与卡通的结合技巧

本节以游戏片断程序设计为例，介绍动态背景与卡通图案的结合技巧。

一、《太空探险》简介

“飞天”号太空飞船欲降落到某个星球上进行科学考察，这个星球的外层空间布满了厚厚的、密密的太空陨石，这些陨石既不降落、也不离去，悬浮在星球的上空。飞船下降必然要与这些陨石相撞，这么多的陨石撞击，飞船是承受不了的。通过环绕该星球飞行发现，有一条从空中通往星球表面的圆形通道，通道中的陨石较稀疏，飞船可以绕过陨石而降落。这条通道，取名为“太空通道”。

二、程序设计策划

根据简介内容，可在屏幕的左侧绘制一条上、下贯通的“太空通道”，太空飞船在太空通道的上端发出滴滴声，向下飞速降落，各种形状的陨石随机散布在飞船的两边。用1#操纵器控制飞船左、右移动，以躲避陨石的撞击，计数器记录着飞船降落的距离。若飞船降落1000米没有与陨石相撞则表示着陆成功，打印贺词、停机。若降落过程中，飞船与陨石发生碰撞，计数器则从零开始。

三、程序清单

根据以上策划编写的程序见 No. 2－17。

```
5  REM No. 2－17 TAI KONG TAN XIAN
10  CLS :SPRITE ON :CGSET 0,1
20  PLAY "V15Y2T2:V15Y3T2"
30  DEF MOVE (7)=SPRITE(7,1,1,1,0,2) :
    POSITION 7,80,40:MOVE 7
40  DIM A$(15) :A$(15)=CHR$(199) :B$=CHR
    $(244)
50  FOR I=0 TO 14 :A$(I)=CHR$(207+I) :NEXT
60  LOCATE 0,21 :PRINT A$(RND(16));B$
70  IF D=2 THEN LOCATE RND(15)+2,22 :PRINT A
    $(RND(16)):D=0
80  LOCATE 17,23 :PRINT B$;A$(RND(16))
90  T=T+1 :IF T MOD 10=0 THEN LOCATE 20,20:
    PRINT "SC="T" "
100  IF T=100 THEN PLAY "O3E0DCDE" :ERA 7 :
     CLS:LOCATE 8,10: PRINT "VERY GOOD!!!" :
     LOCATE 0,20 :END
110  D=D+1 :S=0
120  K=STICK(0) :IF (K=1)+(K=2)=0 THEN 160
130  IF K=1 THEN S=3
140  IF K=2 THEN S=7
150  IF F=1 THEN F=0 :GOTO 190
160  PX=(XPOS(7)－16)/8
170  L$=SCR$(PX－1,2) :R$=SCR$(PX+1,2) :F
```

```
    $=SCR$(PX,2)
180 IF L$<>" "OR R$<>" "OR F$<>" "THEN F
    =1 :GOTO 210
185 IF S=0 GOTO 195
190 DEF MOVE(7)=SPRITE(7,S,1,4,0,2) :
    POSITION 7,PX*8+16,40 :MOVE 7
195 PLAY "O2#G0"
200 GOTO 60
210 PLAY "O1E1C:O0#E1C"
220 T=0 :LOCATE 20,22 :PRINT "AGAIN!" :LO-
    CATE 20,20:PRINT "SC= 0 " :GOTO 60
230 END
```

四、程序说明

1. 30 行定义太空飞船。40、50 行选定太空陨石的形状装入数组 A(15)，选定通道边界图形装入 B$。

2. 60 行画通道左边界。70 行随机打印通道中的陨石，引入变量 D 的作用是控制陨石的密度。D 值越大、陨石越少。80 行画通道的右边界。

3. 90、100 行完成记分、分数显示和降落成功后的简单结束处理。

4. 120～140 行读操纵器的方向键值。

5. 150～180 行判断太空船是否与通道边界或陨石相撞。若相撞则发声，记数从零开始；若没有相撞则执行游戏者的命令，飞船左、右移动，并发出滴滴声。

6. 185～200 行使飞船运动。

7. 210～220 行使记数从零开始。

运行程序，现在你就是一名飞船驾驶员，愿你能沉着冷静地驾驶飞船、机智灵活地躲避陨石、安全平安地降落着陆。

本程序短小精悍，自成游戏片断；情节激烈有趣，富有冒险精神。若再配以开始、结束画面和难度选择等处理程序，将构成一个完整的探险类游戏。

第四节　神奇画笔

本节给出一个功能强大的使用操纵器控制绘图的 F BASIC 绘图程序——神奇画笔。

之所以取名“神奇”，必有其出神入化、奇妙无伦之处。下面我们以使用说明、程序清单、程序说明三个部分介绍这一软件。

一、使用说明

运行程序后，屏幕上将显示一个充满屏幕的“神”字，下方的“□”即为光标。此时，按“Home”键即可清屏，使用游戏主操纵器即可以随意绘图。

下面分基本功能和特殊功能两部分介绍绘图操作方法。

基本功能：

1. 移动画笔

使用主操纵器的“十”字方向键，可控制光标在屏幕上向八个方向运动。

2. 绘制图形

按操纵器的“A”键，可在光标处绘出选定的图形。按住“A”键不放，再按“十”字键往任意方向走，即可画出连续的各种线段。

3. 擦除图形

移动光标盖住欲擦除的图形，同时按“A、B”键即可擦除光标下的图形。

4. 图形配色

按“B”键即可对光标下的图形配色。配色操作中要注意两点：第一要把握速度，及时松开按键，否则四种配色将变换不停；第二要注意染色图形周围的图形搭配，避免把不需要配色的图形染色（因一个图形配色将影响与其相邻的四个图形）。

5. 选择图形

当欲改变绘图用的图形块时，可按“START”键。按键时，屏幕下方供选择的图形符号将不停的向右移动变换，在中间由四个图块拼成的 16×16 点阵的大图形即为选中的图块。选定图形后，松开“START”键，即可按“A”键使用新的图块绘图。

若想向左移动选择图块，可按“SELECT”键，操作方法同向右移动图形。

若想快速移动图形选图，可同时按住“START”、“SELECT”两键，这时图形将以行为单位翻页。

6. 滑稽音响

按一下“十”字键的正中，整个绘画过程将伴随着各种怪响进行，许多声响都是你以前没有听到的。若要关闭滑稽音响，按任一功能键即可。

7. 键入字符

键盘上的所有可输入的字符都可随时键入，显示于光标指向的位置。这一功能可极方便地在画面上标注文字。

8. 清除屏幕

按“Home”键即可清屏再绘，所绘图形将存入图形记忆区，可随时再现。

9. 重新绘画

若感到所绘图形不满意，想重新绘图时，可按“Shift＋Home”键，即可清除一切内容(包括图形记忆区的内容)。

特殊功能：

在作图中，使用 F1～F8 八个热键，可以实现坐标跟踪、对称绘图、屏幕纵横移动、自动绘框、图形自由拼贴、随机变色、笔迹追忆七大特殊功能。

注——每种特殊功能服务均按“Esc”键返回主控作图状态。即按热键进入、按“Esc”键退出。

1. 坐标跟踪

按 F1 键或“Ctrl＋A”复合键。

这是一个坐标显示的逻辑开关。按一次将显示

X:m

Y:n

的字样，且随着光标的移动而变化。其中 m、n 即为光标的目前横、纵坐标值。此字样提示不影响作图。当按第二次热键时，将发出一声“BEEP”，然后提示消失。

2. 对称绘图

按 F2 键或“Ctrl＋B”复合键。

进入功能状态后，屏幕下方将提示

KEY 1,2,3(X/Y/Zero?)

提示的意思是：

若需要横向对称绘图按“1”键

若需要纵向对称绘图按“2”键

若需要以原点对称绘图按“3”键

选项后，即进入对称绘图状态。此后再绘图时，将以选定的对称方式绘制图形。这一功能可以最快的方式绘出一些对称的花纹。若不需要对称绘图时，按下热键后再按除“1、2、3”以外的任意键，即可关闭对称绘图。

3. 屏幕纵横移动

按 F3 键或“Ctrl＋C”复合键。

进入功能状态后，屏幕下方提示：

SCREEN MOVE!

此时按“十”字方向键，整个屏幕的画面将缓慢的向所指定的方向移动。由此可以体会任天堂游戏画面上下滚动、左右卷动、斜向移动的各种变化过程，若在 F BASIC 显示页绘上图形，将观察到两幅画面连接移动的情景。

4. 自动绘框

按 F4 键或“Ctrl＋D”复合键。

进入功能状态后，屏幕下方提示：

SQUARE　ON

此时首先选好绘框所用的图形，然后按“A”、或“B”键，将在光标处出现“＜”或“＞”标志。当此标志形成一定的倾斜度时，按空格键，可立即在屏幕上用所选符号绘出以“＜”、“＞”号为对角顶点的矩形框。绘图速度之快将出乎你的意料。

使用该功能也可以绘出长条或点，本程序的封面“神”字就是调用此模块绘出的。

5. 图形自由拼贴

按 F6 键或“Ctrl＋F”复合键。

进入功能状态后，屏幕下方提示：

FRE DEF ON

这时，可进行图形的任意拼贴。

我们知道，在 BG GRAPHIC 键控绘图中，有 MOVE、COPY 两项功能，但使用时只能逐个图形小块搬移、拷贝，操作极不方便。“神笔”中，这些操作轻而易举，极其简单，而且还可以任意自由组合 8 个以内的“自由图形块”，效果颇佳。

图形自由拼贴的具体操作是：

按 STOP 键，即可将光标处的图符定义到“块”中，移动光标到任意处，可以拼出各种长条块。“块”显示在屏幕的左上方，最多可定义 8 个图形（注意不要贪多，图形多于 8 个时，所定义的块将全部清除）。

“块”定义完毕，移动光标到复制图形的位置，按“A”键即可把“块”复制到光标处。这样可以大幅度的绘画了。若按“B”键，将把“块”中的图符以光标为顶点纵向排列。按 SELECT 键将清除“块”中的内容，可定义新“块”。

6. 随机变色

按 F7 键或“Ctrl+G”复合键。

受 BS 版本内存容量的限制，本程序摒弃了逐块配色的操作，采用了随机配色方式。实践证明，以这种方式操作，所配出的色彩更亮丽，操作最简单。

进入功能状态后，屏幕下方提示：

COLOR CHANGE

此时按住“十”字键的“下”，可以变化屏幕的底背景色；按“上”则变化图符的颜色。注意，配色效果只有按“A”键时才能显现出来。按 F1～F4 中的任意键可使光标下的图符按 0～3 号颜色变化。

配色满意后，按“B”键即可在屏幕下方获得配色数据，可记录下来留待编程使用。若配色不满意或配的乱七八糟，按 START 键即可复原。

7. 笔迹追忆

按 F8 键或“Ctrl＋H”复合键。

此功能为“神笔”的得意之处。电视连续剧《包青天》中，电视剧名逐笔写出的效果可能给你留下深刻的印象。在“神笔”中完全可以模仿这种效果，画出你的绘图轨迹。

进入功能状态后，当屏幕下方提示：

RECOLLECT・・・

时，屏幕上方将飘来一支羽毛笔，这时屏幕上所绘的图形将被清除。

按住“A”键不放，再按“上”或“左”键，即可看到羽毛笔以倒叙的方式把你所绘图形复绘一遍。如果松开“A”键，羽毛笔即以“抬笔”方式，凌空比划所绘图画的轨迹。

若按住“A”键不放，再按“下”或“左”键，羽毛笔将以顺叙追忆你的绘图过程。在追忆过程中，屏幕的右上角有一个数字提示对应的笔划号码。运用“抬笔”和“A”键，你可以轻而易举地实现对局部、片断画面的回忆。

最省力、最精彩的追忆是按“十”字键的中央了，此时你将看到羽毛笔行云流水般的挥动，你所绘制的美妙作品将重新展现在你的面前。

二、程序清单

神奇画笔的程序清单见 No. 2－18。

```
5  REM No. 2－18 SHEN QI HUA BI
10 REM B? EA? CIEEEGSBGDIAICK@LBNHGIJ
   KBWNMDPGRDUGMIPLRI ULP? RBPNRS
20 CLS :CGSET 1,1 :CGEN 3 :SPRITE ON :C=63 :W
   =255:R=184 :DIM M(3) :LOCATE 9,22 :PRINT "
```

```
    deft brush"
30  FOR M=28738 TO 28793 STEP 4 :FOR I=0 TO 3 :
    M(I)=PEEK(M+I)-C :NEXT :K=2 :GOSUB 350
    :GOSUB 420 :NEXT
40  FOR I=1 TO 8 :KEY I,CHR$(I) :NEXT :GOTO
    190
110  IF X<16 THEN X=232
112  IF X>232 THEN X=16
114  IF Y<24 THEN Y=192
116  IF Y>192 THEN Y=24
118  SPRITE 0,X,Y :IF S>0  THEN A=X/8-2 :B=
     Y/8-3
120  RETURN
130  S=STICK(0) :K=STRIG(0) :E$=INKEY$ :E=
     ASC(E$) :W=255
132  IF S=1 THEN X=X+T
134  IF S=2 THEN X=X-T
136  IF S=4 THEN Y=Y+T
138  IF S=8 THEN Y=Y-T
140  IF S=5 THEN Y=Y+T :X=X+T
142  IF S=6 THEN Y=Y+T :X=X-T
144  IF S=9 THEN Y=Y-T :X=X+T
146  IF S=10 THEN Y=Y-T :X=X-T
148  IF S=15 THEN POKE 16405,1
150  IF X>W THEN X=0
152  IF X<0 THEN X=W
154  IF Y>W THEN Y=0
```

```
156 IF Y<0 THEN Y=W
158 IF G=1 THEN GOSUB 110
160 IF S>0 OR K>0 THEN POKE 16384,M,M,M,M :
    M=RND(256):IF S>0 THEN M=154+S
170 IF E=11 THEN RUN
172 IF E=12 THEN CLS
174 IF E=27 THEN 200
176 IF E=26 THEN LOCATE 0,22 :PRINT "BYE-
    BYE?" :IF INKEY$(0)="Y" THEN SPRITE OFF
    :END
180 RETURN
190 CLEAR :DIM M(3) :BK=24601 :BA=BK :BN=
    28000 :DEF SPRITE 0,(0,0,0,0,0)=CHR$(176)
200 CGEN 3 :G=1 :T=8 :R=220 :C=1 :L=0 :D=0
202 LOCATE 0,22 :PRINT "DEFT BRUSH-----
    -" :GOSUB 350
210 GOSUB 130 :IF E>31 THEN LOCATE A,B :
    PRINT E$
212 IF K>0 AND K<4 THEN GOSUB 350
214 IF E>0 AND E<9 THEN LOCATE 2,22 :BEEP :
    ON E GOSUB260,300,340,400,440,550,600,700
216 IF K=4 THEN COLOR A,B,C :C=C+1 :M=138 :
    IF C>3 THEN C=0
218 IF K=12 THEN LOCATE A,B :PRINT " " :M=
    129
220 IF K=8 THEN LOCATE A,B :PRINT R$ :IF S>0
    THEN POKE BA,A,B,R :BA=BA+3 :IF BA>BN
```

```
    THEN BA=BK
222 IF L>0 AND S>0 THEN GOSUB 280
224 IF D>0 AND K=8 THEN GOSUB 320
250 GOTO 210
260 PRINT "LOCATE ON"; :L=L+1 :IF L=2 THEN
    L=0 :PRINT "OFF!" :BEEP
270 GOTO 202
280  DEF SPRITE 1,(3,1,0,0,0)="X:Y:" :DEF
     SPRITE 7,(3,1,0,0,0)=MID$(STR$(A)+" ",
     2,2)+MID$(STR$(B)+"  ",2,2) :SPRITE 1,
     X,Y+8 :SPRITE 7,X+16,Y+8
290 RETURN
300 D=D+1 :IF D>1 THEN D=0
302 PRINT "KEY 1,2,3:(X/Y/Z?)"; :F=VAL(INKEY
    $(0)) :IF D=0 OR F>3 OR F<1 THEN PRINT "
    OFF" :F=0
310 GOTO 202
320 IF F=1 THEN LOCATE 27-A,B :PRINT R$
322 IF F=2 THEN LOCATE A,21-B :PRINT R$
324 IF F=3 THEN LOCATE 27-A,21-B :PRINT R$
330 RETURN
340 PRINT "SCREEN MOVE!" :G=0:T=2
342 GOSUB 130 :POKE 8917,X :POKE 8917,Y
345 GOTO 342
350 IF K=1 THEN R=R-1
352 IF K=2 THEN R=R+1
354 IF K=3 THEN R=R+7
```

```
356 IF R<33 THEN R=W
358 IF R>W THEN R=33
360 FOR I=1 TO 7 :R1=R+I-4 :IF R1>W THEN R1
    =33+I
362 DEF SPRITE I,(0,0,0,0,0)=CHR$(R1) :SPRITE
    I,I*30,220 :IF I>4 THEN SPRITE I,I*30,212
364 NEXT :R$=CHR$(R) :DEF SPRITE 4,(0,1,0,
    0,0)=R$+R$+R$+R$ :SPRITE 4,120,210
370 RETURN
400 PRINT "SQUARE ON"
402 GOSUB 130 :IF K<4 AND K>0 THEN GOSUB 350
404 IF K=4 THEN DEF SPRITE 1,(3,0,0,0,0)="C" :
    SPRITE 1,X,Y :M(0)=A :M(1)=B :M=175
406 IF K=8 THEN DEF SPRITE 7,(3,0,0,0,0)="}" :
    SPRITE 7,X,Y :M(2)=A :M(3)=B :M=175
408 IF E=32 OR E=13 THEN S=15 :M=175 :GOSUB
    148 :GOSUB 420
410 GOTO 402
420 L=1 :IF M(2)<M(0) THEN L=-1
422 FOR I=M(0) TO M(2) STEP L :LOCATE I,M(1)
    :PRINT R$:LOCATE I,M(3) :PRINT R$ :
    NEXT
424 L=1 :IF M(3)<M(1) THEN L=-1
426 FOR I=M(1) TO M(3) STEP L :LOCATE M(0),I
    :PRINT R$:LOCATE M(2),I :PRINT R$ :
    NEXT
430 RETURN
```

```
440  PRINT "SPRITE PEN" :BEEP :PRINT "ERA!"
450  CGSET 1,1 :RETURN
500  PRINT "COLOR CHANGE" :CGSET 0,1
502  GOSUB 130 :IF S=4 THEN G=RND(W)
504  IF S=8 THEN C=RND(W) :D=RND(W) :F=
     RND(W)
508  IF K=1 THEN CGSET 1,1
510  IF E<5 AND E>=1 THEN R=E-1 :BEEP
512  IF K=8 THEN PALETB R,G,C,D,F
514  IF K=4 THEN CGSET 1, 1 :LOCATE 0, 22 :
     PRINT "PALETB";R;C;D;F :PAUSE :K=8
520  GOTO 502
550  PRINT "FRE DEF!" :Q$="" :S$="" :L=1
552  GOSUB 130 :IF K=1 THEN R$=SCR$(A,B) :S
     $=S$+R$ :DEF SPRITE L,(0,1,0,0,0)=S$
     +"——" :SPRITE L,L*16,10
     :Q$=Q$+R$ :BEEP
554  IF S=0 THEN FOR I=1 TO L :SPRITE I,I*16=
     8,8-I :NEXT
556  IF LEN(S$)>1 THEN L=L+1 :S$=""
558  IF K=4 THEN FOR I=1 TO LEN(Q$) :IF I+B<
     22 THEN LOCATE A,B+I-1 :PRINT MID$(Q
     $,I,1) :NEXT
560  IF K=8 THEN LOCATE A,B :PRINT Q$
562  IF K=2 OR L>5 THEN L=1 :Q$=""
570  GOTO 552
600  PRINT "COLOR CHANGE" :CGSET 0,1
```

```
602 GOSUB 130 :IF S=4 THEN G=RND(W)
604 IF S=8 THEN C=RND(W) :D=RND(W) :F=
    RND(W)
608 IF K=1 THEN CGSET 1,1
610 IF E<5 AND E>=1 THEN R=E-1 :BEEP
612 IF K=8 THEN PALET B R,G,C,D,F
614 IF K=4 THEN CGSET 1,1 :LOCATE 0,22 :PRINT
    "PALETB";R;C;D;F :PAUSE :K=8
620 GOTO 602
700 PRINT "RECOLLECT..." :CGEN 3 :S$=""  :G=
    0 :FOR I=232 TO 239 :S$=S$+CHR$(I) :
    NEXT
702 DEF SPRITE 6,(2,1,0,0,0)=MID$(S$,5,4) :
    DEF SPRITE 7,(2,1,0,0,0)=MID$(S$,1,4)
704 SPRITE 0 :CLS :CGEN 2 :FOR I=W TO 28 STEP
    -1 :SPRITE 7,I,20 :SPRITE 6,I,36 :NEXT
706 F=BA-BK :L=F
710 GOSUB 130 :C=PEEK(F+BK) :D=PEEK(F+BK
    +1) :IF S>0 THEN GOSUB 730
712 IF S=8 OR S=2 THEN F=F-3 :IF F<0 THEN F
    =L
714 IF S=4 OR S=1 THEN F=F+3 :IF F>L THEN F
    =0
716 IF K=4 THEN LOCATE C,D :PRINT " " :M=129
718  IF K=8 THEN LOCATE C,D :PRINT CHR
     $(PEEK(BK+F+2)) :LOCATE 20,22 :PRINT
     F/3;"B."
```

```
720  IF S=15 OR E=66 THEN M=175 :GOSUB 148 :
     FOR I=BK TO BA-3 STEP 3 :C=PEEK(I) :D=
     PEEK(I+1) :GOSUB 730:LOCATE C,D :PRINT
     CHR$(PEEK(I+2)) :NEXT
725  GOTO 710
730  R=C*8+16 :M=D*8 :SPRITE 7,R,M :SPRITE
     6,R,M+16
731  RETURN
```

三、程序说明

1. 10～30 行系统初始化，绘制标题画面的大“神”字。其中 10 行使用 REM 语句给出了大“神”字各笔划的绘图坐标值，仅以 56 个字符表示了 28 条“LOCATE”语句，这在 BASIC 语言编程中是非常巧妙之作。

注意——输入 10 行语句时，行号、指令、注释内容之间不能留空格。

2. 40 行定义特殊绘图功能的 8 个热键。

3. 110～120 行是绘图控制中，确定绘图坐标的子程序。

4. 130～180 行是光标及部分特殊功能的控制子程序。

5. 190～214 行为作图主控程序对各子模块的调用控制，其中使用“ON”语句简化了转移操作语句。

6. 216～250 行完成对图形配色、绘图绘制、擦除图形的功能。

7. 260～290 行是绘图坐标跟踪控制子程序。坐标提示通过卡通实现。

8. 300～330 行完成对三种对称绘图过程的控制。

9. 340～345 行完成了显示窗口各方向移动的控制。

10. 350～370 行为作图符号选择控制子程序。

11. 400～430 行为自动绘制图框子程序。其中以两组循环实现了任意长宽比构成的方框。

12. 500～520 行是随机置色子程序，若想查阅数据，按“B”键实现，查阅后按任意键返回。

13. 550～570 行为图形自由拼贴功能子程序。

14. 700～731 行为绘图追忆子程序。

注——若程序运行后显示

? OM ERROR IN ×××

表示内存不足，可考虑删除某段功能子程序。

绘图结束后，当不慎使用“CLS”指令清除了屏幕或使用“NEW”清除了程序，使用下列程序仍可恢复所绘的图形。

```
1000  CLS :CGSET 1,1 :CGEN 3
1010  FOR I=24601 TO 28000 STEP 3
1020  X=PEEK(I) :Y=PEEK(I+1) :C=PEEK(I+2)
      ::IF X>27 THEN END
1030  LOCATE X,Y :PRINT CHR$(C)
1040  NEXT
```

四、编程特点

下面简要总结本程序的编程特点：

1. 使用 REM 语句存储游戏封面的数据是一个创举，可以大量节约有限的用户内存。

2. 绘图轨迹中引入抬笔的概念，既便于任意图画片断的回忆，又能锻炼人们的空间联想力。

3. 使用两点定位绘制图框，方法巧妙，速度惊人。

4. 画面任意位移，充分展现了 6527 系统的图形处理控

制能力。

5. 通过控制系统发声软开关，发出多种效果音响，充分利用了 6527 系统的声音功能。

总之，《神奇画笔》是一个构思奇巧的绝妙程序，除卡通功能外，它已经把 F BASIC 的图形、声响功能发挥的淋漓尽至，是一个不可多得的程序。

有关 F BASIC 编程绘图类的游戏，还有许多，我们将把它们放在丛书之九《BASIC 程序集》中介绍。

第三章　机器语言绘图基础

任天堂游戏向人们展现出一片神奇的图画天地：里面有嵯峨的群山、幽冥的星空、葳蕤的丛林、辉煌的迷宫……；有迷人的草原丽景、有绝妙的海底龙宫、有令人叹为观止的沙漠风光、有气势如虹的未来战争……。这些精彩逼真、内容丰富的画面，引起千百万人苦苦思索，弄不清楚同是8位机的任天堂，为什么能展现出效果远远优于苹果机、中华学习机的绚丽画卷？其中的奥妙在哪里呢？

奥秘的焦点集中在硬件结构上。

任天堂游戏机的硬件系统中采用了“图象处理器”——PPU，专门用于承担图象显示任务。CPU对PPU的控制仅通过8个系统软开关完成（见丛书之六），这样使CPU从大量的图形数据处理中脱出身来，集中精力完成游戏进程的控制和音乐演奏。从而使背景画面、卡通图案、音响三大系统同步工作，使一个CPU发挥了两个CPU的作用。

本章将以通俗的叙述，介绍任天堂游戏机6527CPU显示系统的屏幕结构、背景画面的显示原理、卡通画面的显示原理、屏幕显示状态的设置、CPU与PPU的通信方法、色彩配置原理及6527CPU的中断系统等内容。

第一节　6527CPU的显示屏幕结构

任天堂游戏精彩画面的实现，取决于独特的屏幕显示结

构，使背景画面、卡通图案等各种处理快捷而简便。

6527CPU 的显示屏幕由四个显示页面叠加而成，自前向后依次为卡通 0 页、背景页、卡通 1 页、底背景页。各页的排列情况如图 3—1 所示。

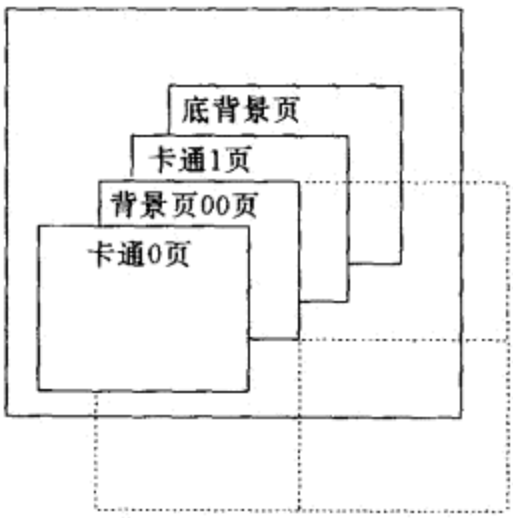


图 3—1　6527CPU 显示页面结构图

各个页面的显示可通过系统软开关灵活地控制。在显示图象时，靠前显示页上的内容可以覆盖后面显示页上的内容。因此，通过显示页面间的协调配合可以形成具有一定景深次序的、立体感极强的三维画面。

第二节　背景显示画面的结构

背景页位于卡通 1 页之前、卡通 0 页之后，从图 3—1 可见，它由四个显示页面构成。它们的默认排列位置如图 3—2 所示。

正常显示时，屏幕显示背景 00 页的内容，工作中可通过系统软开关的设置显示任一页。还可以由系统软开关控制，以点为单位实现两个显示页面的左右卷绕或上下翻滚。

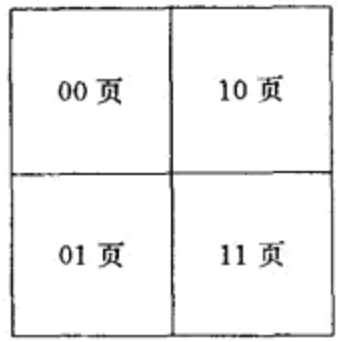


图 3—2　背景页面排列示意图

一、显示元素与 PPU 地址的对应关系

在四个背景页面中，每个背景页面都可以显示背景图案或各种字符，每页可显示 32 列×30 行共 960 个图形块，我们把每个图块位置称为“显示元素”。960 个显示元素由 PPU 直接控制，进行定格显示，每个显示元素与 PPU 地址的关系为顺序对应。

1. 背景 00 页

背景 00 页的显示，使用了 PPU 所管理内存的 $2000～$23BF 共 960 个单元，每个显示行与 PPU 地址的对应关系如表 3—1 所示。

表 3—1　背景 0 页(00 页)各显示行的对应地址一览表

行　次	PPU　地　址	行　次	PPU　地　址
一	2000 ～ 201F	十六	21E0 ～ 21FF
二	2020 ～ 203F	十七	2200 ～ 221F
三	2040 ～ 205F	十八	2220 ～ 223F
四	2060 ～ 207F	十九	2240 ～ 225F
五	2080 ～ 209F	二十	2260 ～ 227F
六	20A0 ～ 20BF	廿一	2280 ～ 229F

续表

行　次	PPU　地　址	行　次	PPU　地　址
七	20C0 ～ 20DF	廿二	22A0 ～ 22BF
八	20E0 ～ 20FF	廿三	22C0 ～ 22DF
九	2100 ～ 211F	廿四	22E0 ～ 22FF
十	2120 ～ 213F	廿五	2300 ～ 231F
十一	2140 ～ 215F	廿六	2320 ～ 233F
十二	2160 ～ 217F	廿七	2340 ～ 235F
十三	2180 ～ 219F	廿八	2360 ～ 237F
十四	21A0 ～ 21BF	廿九	2380 ～ 239F
十五	21C0 ～ 21DF	三十	23A0 ～ 23BF

2. 背景 01 页

背景 01 页的显示使用 $2400～$27BF 共 960 个单元。每个显示行与 PPU 地址的对应关系如表 3—2 所示。

表 3—2　背景 1 页(10 页)各显示行的对应地址一览表

行　次	PPU　地　址	行　次	PPU　地　址
一	2400 ～ 241F	十六	25E0 ～ 25FF
二	2420 ～ 243F	十七	2600 ～ 261F
三	2440 ～ 245F	十八	2620 ～ 263F
四	2460 ～ 247F	十九	2640 ～ 265F
五	2480 ～ 249F	二十	2660 ～ 267F
六	24A0 ～ 24BF	廿一	2680 ～ 269F
七	24C0 ～ 24DF	廿二	26A0 ～ 26BF
八	24E0 ～ 24FF	廿三	26C0 ～ 26DF
九	2500 ～ 251F	廿四	26E0 ～ 26FF
十	2520 ～ 253F	廿五	2700 ～ 271F

续表

行　次	PPU　地　址	行　次	PPU　地　址
十一	2540 ～ 255F	廿六	2720 ～ 273F
十二	2560 ～ 257F	廿七	2740 ～ 275F
十三	2580 ～ 259F	廿八	2760 ～ 277F
十四	25A0 ～ 25BF	廿九	2780 ～ 279F
十五	25C0 ～ 25DF	三十	27A0 ～ 27BF

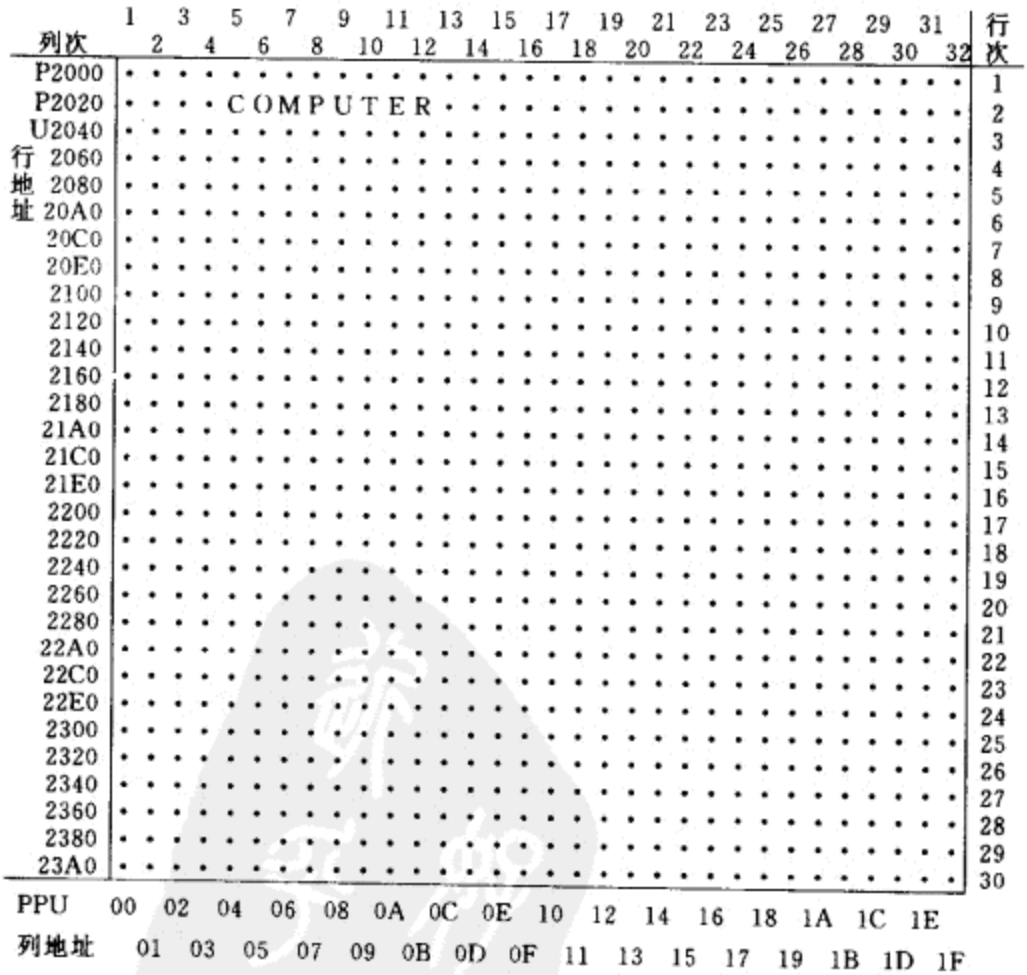


图 3-3　背景 0 页(00 页)显示单元与 PPU 地址对应图

3. 背景 10、11 页

背景 10 页的显示使用 $2800～ $2BBF 共 960 个单元。

背景 11 页的显示使用 $2C00～$2FBF 共 960 个单元。每个显示行与 PPU 地址的对应关系与 00、01 页类似，读者可自己排列。

4. **显示元素与 PPU 地址的对应关系**

每个显示元素与 PPU 地址一一对应，背景 00 页的对应关系见图 3－3，背景 01 页的对应关系见图 3－4；其它两个页面的对应规律与上面两页相同。

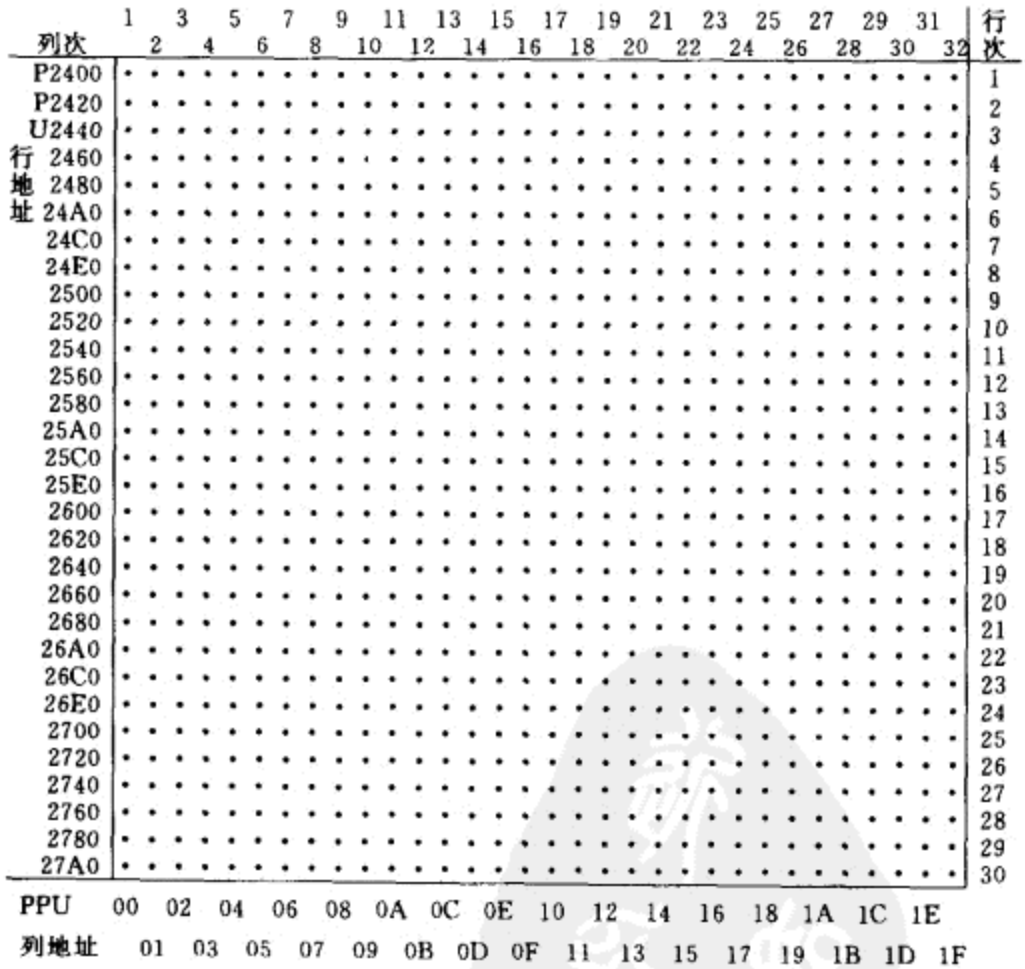


图 3－4　背景 1 页(10 页)显示单元与 PPU 地址对应图

(1)图中的列次表示图形块或字符位于显示屏幕的第几列，行次则表示图形块或字符位于显示屏幕的第几行。如“COMPUTER”的第一个字母“C”则位于第四列第二行。

(2)每个显示元素所对应的 PPU 行地址加 PPU 列地址即为该显示元素所对应的 PPU 地址。如字母“C”的行地址为 2020H、列地址为 03H，那么“C”的 PPU 地址则为 2023H，记为 $2023。其它依次类推。

4. 显示页面与 VRAM 的对应关系

由于 PPU 的 VRAM 只有 2K 空间，所以 CPU 自动设定 00、01 页映射到同一 VRAM 单元。即对 00 页显示单元置入数值时，可在 00、01 两页同时显示。同理，当对 01 页显示单元 $2800～$2BFF 置入数值时，也可在 00、01 两页同时显示。同样，10、11 两页也映射到同一 VRAM 单元。因此，通常把 00、01 页称为背景 0 页，把 10、11 页称为背景 1 页。特殊需要时，可用软开关把两个页面分开，写入不同的数据以显示不同的图形。

二、显示单元的配色

为叙述方便，我们把屏幕上的一个图象元素(一个 8×8 点阵的方块)称为一个显示单元，它是背景显示页面的最小显示单位，也可以称为一个图形块。由于每一个显示单元可以显示一个图形块，那么每个显示单元也应该有其确定的配色。就因为 6527CPU 可以方便灵活地控制每个显示单元的配色，所以才显示出五彩缤纷的绚丽画面。

1. 配色特点

6527CPU 的配色方法独具特色，它既操作简单又变化千般，是诸多微机系统中较优化的配色方法。它的特点如下：

(1)它把背景显示页的 32 列×30 行 960 个显示单元按确定的格式进行新的排列组合，使每相邻的四个显示单元为一组，把整幅画面分成 16 列×15 行共 240 个小方格(第 15

行每格只有两个显示单元)。每个小方格有四组颜色供选择，这四组颜色对应着 F BASIC 语言中背景配色板中的四个配色代码。由于每个配色代码由三种颜色代码组成，故每个小方格有 12 种颜色供选用(若包括底背景配色，实际有 16 种颜色。)每个小方格的四种配色代码由两位二进制数表示(即显示画面中每相邻的四个单元用一种配色代码)。

(2)再把 16 列×15 行的 240 个小方格，以上下左右相邻的四个小方格为单位组成一个大方格，即每个大方格由四个小方格、16 个显示单元组成，其配色代码则由 8 位二进制数表示，刚好占用一个配色单元。每个小方格的配色代码在配色单元中(8 位二进制数)的排列方式如图 3—5 所示。

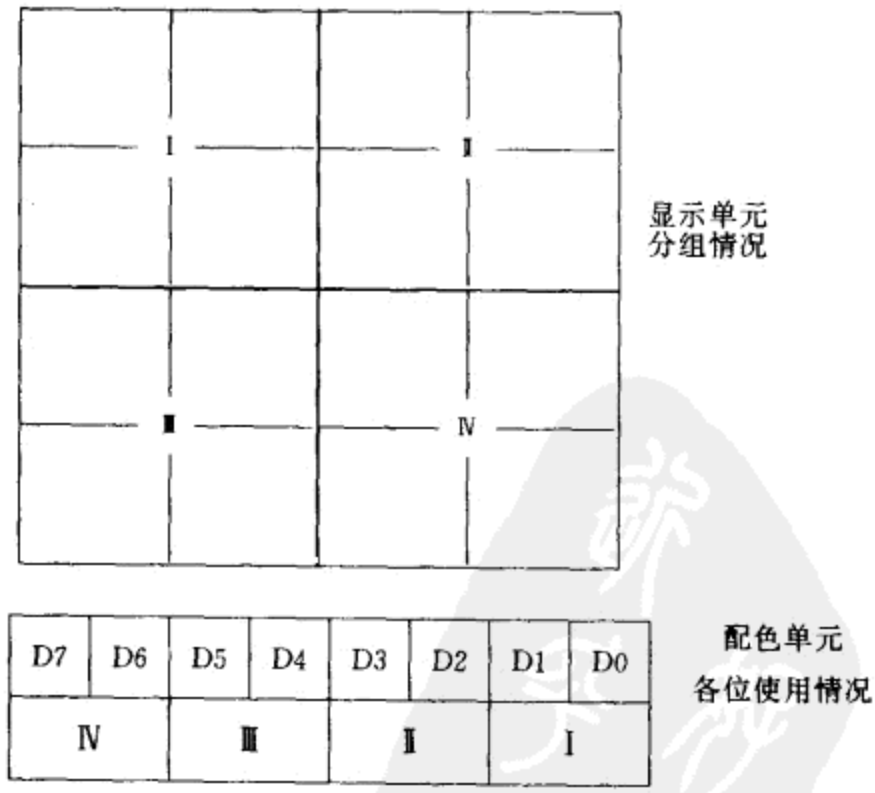


图 3-5　显示单元配色分组及配色代码排列示意图

(3)通过 1.、2. 两步的处理，把一个显示页面分成 8×8 列 64 个大方格，共占用 64 个配色单元。

2. 配色地址与显示单元的对应关系

背景 00 页的配色占用 $23C0～$23FF；背景 10 页的配色占用 $27C0～$27FF；01 页的配色占用 $2BC0～$2BFF；11 页占用 $2FC0～$2FFF。为便于使用，特把 00、10 页的显示单元与配色地址的对应关系示于图 3－6、图 3－7。

注——图中使用了两种符号，以区别相邻的配色单元。

当欲改变某显示单元的配色时，只要把配色数据写入相应的配色单元就可达到目的。因此，6527CPU 显示配色的设置操作是十分简单方便的。

列次	1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32	行次
P2000	+ + + + · · · · + + + + · · · · + + + + · · · · + + + + · · · ·	1
P2020	+ 23C0 + · 23C1 · + 23C2 + · 23C3 · + 23C4 + · 23C5 · + 23C6 + · 23C7 ·	2
U2040	+ + + + · · · · + + + + · · · · + + + + · · · · + + + + · · · ·	3
行 2060	+ + + + · · · · + + + + · · · · + + + + · · · · + + + + · · · ·	4
地 2080	· · · · + + + + · · · · + + + + · · · · + + + + · · · · + + + +	5
址 20A0	· 23C8 · + 23C9 + · 23CA · + 23CB + · 23CC · + 23CD + · 23CE · + 23CF +	6
20C0	· · · · + + + + · · · · + + + + · · · · + + + + · · · · + + + +	7
20E0	· · · · + + + + · · · · + + + + · · · · + + + + · · · · + + + +	8
2100	+ + + + · · · · + + + + · · · · + + + + · · · · + + + + · · · ·	9
2120	+ 23D0 + · 23D1 · + 23D2 + · 23D3 · + 23D4 + · 23D5 · + 23D6 + · 23D7 ·	10
2140	+ + + + · · · · + + + + · · · · + + + + · · · · + + + + · · · ·	11
2160	+ + + + · · · · + + + + · · · · + + + + · · · · + + + + · · · ·	12
2180	· · · · + + + + · · · · + + + + · · · · + + + + · · · · + + + +	13
21A0	· 23D8 · + 23D9 + · 23DA · + 23DB + · 23DC · + 23DD + · 23DE · + 23DF +	14
21C0	· · · · + + + + · · · · + + + + · · · · + + + + · · · · + + + +	15
21E0	· · · · + + + + · · · · + + + + · · · · + + + + · · · · + + + +	16
2200	+ + + + · · · · + + + + · · · · + + + + · · · · + + + + · · · ·	17
2220	+ 23E0 + · 23E1 · + 23E2 + · 23E3 · + 23E4 + · 23E5 · + 23E6 + · 23E7 ·	18
2240	+ + + + · · · · + + + + · · · · + + + + · · · · + + + + · · · ·	19
2260	+ + + + · · · · + + + + · · · · + + + + · · · · + + + + · · · ·	20
2280	· · · · + + + + · · · · + + + + · · · · + + + + · · · · + + + +	21
22A0	· 23E8 · + 23E9 + · 23EA · + 23EB + · 23EC · + 23ED + · 23EE · + 23EF +	22
22C0	· · · · + + + + · · · · + + + + · · · · + + + + · · · · + + + +	23
22E0	· · · · + + + + · · · · + + + + · · · · + + + + · · · · + + + +	24
2300	+ + + + · · · · + + + + · · · · + + + + · · · · + + + + · · · ·	25
2320	+ 23F0 + · 23F1 · + 23F2 + · 23F3 · + 23F4 + · 23F5 · + 23F6 + · 23F7 ·	26
2340	+ + + + · · · · + + + + · · · · + + + + · · · · + + + + · · · ·	27
2360	+ + + + · · · · + + + + · · · · + + + + · · · · + + + + · · · ·	28
2380	· · · · + + + + · · · · + + + + · · · · + + + + · · · · + + + +	29
23A0	· 23F8 · + 23F9 + · 23FA · + 23FB + · 23FC · + 23FD + · 23FE · + 23FF +	30
PPU	00 02 04 06 08 0A 0C 0E 10 12 14 16 18 1A 1C 1E	
列地址	01 03 05 07 09 0B 0D 0F 11 13 15 17 19 1B 1D 1F	

图 3-6　背景 00 页显示单元与配色地址的对应关系示意图

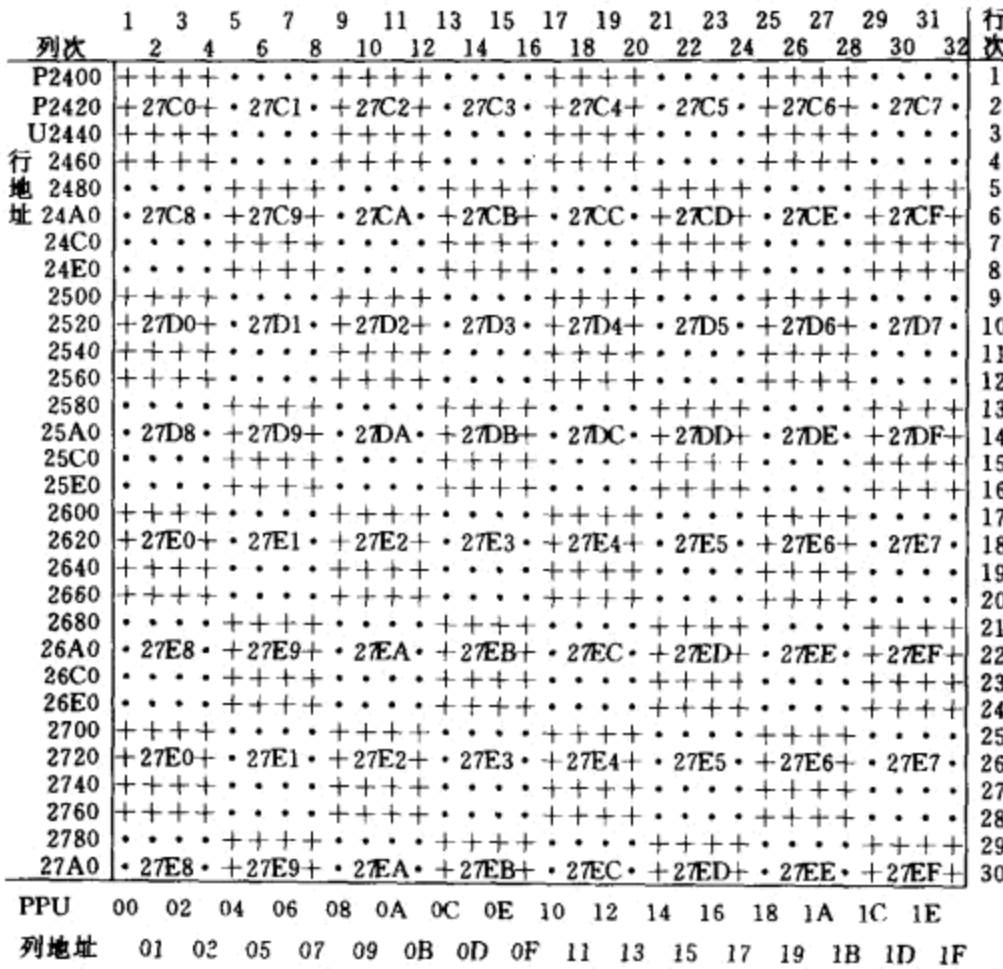


图 3-7　背景 10 页显示单元与配色地址的对应关系示意图

第三节　卡通显示画面的结构

卡通显示画面即是游戏中角色所使用的显示页面，由 0 页和 1 页组成。显示在 0 页上的角色可以遮住背景物，相当于卡通在背景前面活动；显示在 1 页上的角色则相反，可被背景物所遮住，相当于卡通在背景后面活动。因此，在动画处理中，不须象其它微机那样对背景画面点阵进行复杂的逻辑运算就能达到十分理想的效果。

一、卡通页的显示特点

卡通页最多可同时显示 64 个 8×8 点阵角色(卡通块)，在同一水平方向上最多可显示 8 个 8×8 点阵的角色。角色的活动范围为 256×240 点，每次最少可移动一点。可任意指定每个角色左右翻转、上下颠倒等显示状态和四种配色代码的变换。

二、卡通的显示区

卡通页的显示区，一般映射在 CPU 所管理内存的第二页中的 256 个单元(也可通过系统软开关选择任意一页)。这样设计有利于对卡通运动控制的快速处理。在卡通显示定义区中，如在 $0200～$02FF 中，以四个连续单元为一体定义一个 8×8 点阵的卡通块。因此，卡通定义区最多可定义 64 个卡通块。

在 F BASIC 语言中，一般由四个卡通块组成一个 16×16 点阵的角色，所以系统规定最多可显示 16 个角色(SPRITE，简写 SP)，即 8 个显示 SP、8 个运动 SP。它们对应的二页地址如表 3－3 所示。

表 3－3　F BASIC 中显示 SP、动作 SP 对应地址表

显示 SP 编号	地址	动作 SP 编号	地址
0	200 ～ 20F	0	280 ～ 28F
1	210 ～ 21F	1	290 ～ 29F
2	220 ～ 22F	2	2A0 ～ 2AF
3	230 ～ 23F	3	2B0 ～ 2BF
4	240 ～ 24F	4	2C0 ～ 2CF
5	250 ～ 25F	5	2D0 ～ 2DF
6	260 ～ 26F	6	2E0 ～ 2EF
7	270 ～ 27F	7	2F0 ～ 2FF

三、卡通块的定义

定义一个卡通块要使用二页连续的四个单元，对 64 个卡通块来说，每个对应单元的功能是相同的。下面仅以定义第一个卡通块为例介绍每个单元的含义。(第一个卡通占用 $200～$203 四个单元)

1. $200：卡通块显示纵坐标设定单元。取值范围为 0～255，有效值为 0～239。

2. $201：卡通块在字库中的序号设定单元。它可以取第一区的序号，也可以取第二区的序号。取值范围为 0～255。区号可通过系统软开关设定。

3. $202：卡通块显示状态设定单元。各位的定义如下：

D7	D6	D5	D4	D3	D2	D1	D0

D0、D1：卡通块配色设定位，取值为二进制数 00～11。

D4：卡通类型标志。在 F BASIC 中

D4＝0 表示定义的是显示卡通；

D4＝1 表示定义的是动作卡通。

D5：显示页面标志。(在 F BASIC 中称为显示优先度)

D5＝0，卡通在 0 页显示；

D5＝1，卡通在 1 页显示。

D6：左右翻转标志。

D6＝0 为正常显示；

D6＝1 为左右翻转显示。

D7：上下颠倒标志。

D7＝0 为正常显示；

D7＝1 为上下颠倒显示。

4. $203：卡通显示横坐标设定单元。取值范围为 0～255。

显然，当欲在屏幕上某一个位置显示某个卡通块时，只要向有关地址写入相应的数据就可任意显示。

注——在游戏程序的设计中，卡通没有显示卡通、动作卡通之分，$202 的 D4 位不起作用。

例 3－1 在屏幕的(100，150)位置显示黑体数字“4”。

可置入如下数值：

$200＝&H96(150 的 16 进制数)

$201＝&HF3（卡通块“4”的序号为 243，16 进制为 F3H)

$202＝0　　（显示状态取正常显示，0＃配色）

$203＝&H64　（100 的 16 进制数）

相应的 F BASIC 程序为 No. 3－1。

```
5  REM No. 3－1
10  SPRITE ON
20  POKE 512，150
30  POKE 513，243
40  POKE 514，0
50  POKE 515，100
```

当欲使它左右翻转时，可把 40 句改为

```
40  POKE 514，64
```

就可实现左右翻转。

同样，若欲使其上下颠倒则改为

40 POKE 514，128

若欲使其上下颠倒、左右翻转则改为

40 POKE 514，192

若把程序 No. 3－1 改写为 No. 3－2，将会看到数字“4”反复地显示其四种不同的状态，顺序为：正常显示、左右翻转、上下颠倒、上下左右翻转颠倒。

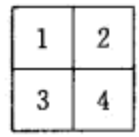
程序 No. 3－2 如下：

```
5  REM No. 3－2
10 SPRITE ON
20 POKE 512，150
30 POKE 513，243
40 POKE 514，A
50 POKE 515，100
60 A = A + 64 : IF A > 192 THEN A = 0
70 PAUSE 50 : GOTO 40
```

若仔细观察将会发现，数字“4”的四种显示状态尽管反复变化，但其右下角的位置恒定不变。这说明 $200，$203 中指定的坐标点作用在卡通块的右下角。这一点，将为我们在程序设计中准确地确定卡通的显示坐标，提供了遵循的准则。

四、角色的定义方法

BS·2A 版本图形库中定义的角色一般为 16×16 点阵，即由四个卡通块组成一个 SP。这四个卡通块的排列顺序如下：



显然，当定义一个卡通时，四个卡通块的坐标是不同的。系统规定：16×16 点阵 SP 的基准坐标仍为第一个卡通块的右下角，其余三个卡通块的坐标则以第一个卡通块的坐标为参考点相应调整。因此，当第一个卡通块的坐标为(100,150)时，第二、三、四号卡通块的坐标依次为(108,150)、(100,158)、(108,158)。即以一号卡通块的坐标为参考，二号卡通块的 Y 坐标不变、X 坐标加 8；三号卡通块的 X 坐标不变、Y 坐标加 8；四号卡通块的 X、Y 坐标都加 8。

现以 0# 显示 SP 的定义为例，介绍各单元数据的设置。

0# 显示 SP 占用 $200～$20F 共 16 个单元，当欲在(96,144)位置显示玛丽哥哥时，则应对各单元置入如下数据(假定玛丽哥哥以正常状态显示、0# 配色)：

$200=90	$204=90
$201=0	$205=1
$202=0	$206=0
$203=60	$207=68
$208=98	$20C=98
$209=2	$20D=3
$20A=0	$20E=0
$20B=60	$20F=68

以上等式中的数据均为 16 进制数

如果我们对 $200～$20F 单元置入上述数据则与执行

F BASIC 程序

```
××  DEF SPRITE 0，(0，1，0，0，0)≡ CHR$
    (0)+CHR$(1)+ CHR$(2)+ CHR$(
    3)
××   SPRITE 0，96，144
```

取得相同的效果。

通过以上的讨论，我们不难想象：任天堂游戏中的大角色(32×32 点阵或 64×64 点阵)无非是使用较多的卡通块组成。如 32×32 点阵角色的定义，只要用 16 个卡通块按照定义 16×16 点阵 SP 的方法设置坐标就可实现(音乐板、计算板、打字板中的羽毛笔就是由 8 个卡通纵向排列组成的)。

实际上，在任天堂游戏程序的设计中，64 个 8×8 点阵的卡通块根本不够用，有时要用到 128 个或更多。解决的方法是：CPU 反复地改写定义卡通块的地址，使几个角色使用同一个映射地址而交替显示。也可以使用再开辟一个卡通定义区，使一个角色的不同部分在两个区中分别定义，通过交替切换卡通定义区实现多个卡通的定义。当这种改写或切换的频率较高时，通过人眼的视觉惰性就能够产生各个角色同时在不同位置显示的效果。但因这样处理后的卡通为间断显示，所以任天堂游戏中大角色卡通存在闪烁现象。

第四节　屏幕显示状态的设置

6527CPU 屏幕显示状态的设置，一般通过两个系统软开关组 $2000、$2001 进行控制。这两个软开关在丛书之六已进行了简单介绍，现在进一步讨论它们的使用方法。

一、状态方式开关组 $2000

$2000 软开关组主要用于控制 6528PPU 的工作状态：由它的各位决定背景页、卡通页显示所使用的图形库区号；控制向 VRAM 写入数据的方式；选定当前显示的背景页面；控制定义卡通的大小；控制中断申请等。各位功能见图 3—8。

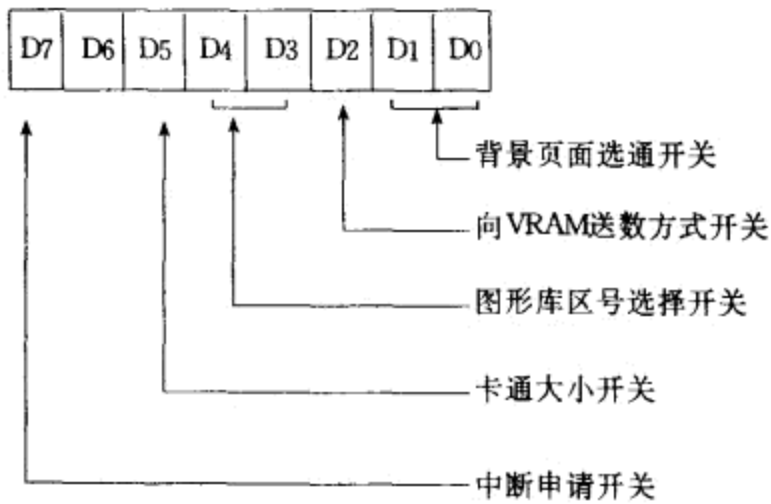


图3—8　工作状态开关组示意图

图 3—8 说明：

1. 背景页面选择开关组 D1、D0 位

D1、D0 两位为背景显示页面的选通开关组，两位二进制数可选通四个显示页面。它们的取值与背景页面的对应关系见表 3—4。

表 3—4　$2000 单元 D1、D0 位与背景页面对照表

D1 位	D0 位	选通的背景显示页面
0	0	00 页
0	1	10 页
1	0	01 页
1	1	11 页

2. 向 VRAM 送数方式选择开关 D2 位

D2 位为向 VRAM 写入数据方式选择开关。当我们向 VRAM 写入数据时，只要向 PPU 置入起始地址，以后可以把欲写入的数据依次送入 VRAM，而不必每写一数据置一次地址，PPU 会控制写入地址自动增加，使地址指针指向下一个置数地址。但是，在写入数据的过程中有一个传数方向问题，表现在屏幕上就是：传送的数据是横向置入在屏幕上显示一行，还是纵向置入在屏幕上显示一列。D2 位就是为实现这一控制而设置的传数方式软开关。

当 D2＝0 时，每送入一个数据，PPU 地址自动加一，表现在屏幕上就是横向显示；

当 D2＝1 时，每送入一个数据，PPU 地址自动加 20H，表现在屏幕上就是纵向显示。

3. 图形库区号选择开关组 D3、D4 位

D3、D4 位为图形库区号选择开关组。其中 D3 位作用于卡通页、D4 位作用于背景页。当它们为 0 时，选中图形库第 I 区的图形，即卡通块的图形，也就是 F BASIC 中符号表 A 的图形；当它们为 1 时，则选中图形库第 Ⅱ 区，即背景字符图形，相当于 F BASIC 中的符号表 B 中的图形。

D3、D4 位共有四种组合，它们刚好对应了 F BASIC 中 CGEN 指令的四种状态，其对应关系见表 3－5。

表 3－5　$2000 单元 D4、D3 位与 CGEN 指令状态对照表

D4 位	D3 位	选中的图形库区号		相对应的 CGEN 指令
		背景页	卡通页	
0	0	I	I	CGEN0
0	1	I	Ⅱ	CGEN1
1	0	Ⅱ	I	CGEN2
1	1	Ⅱ	Ⅱ	CGEN3

通过表 3—5 的对照，我们可以深刻地体会 CGEN 指令的执行实质。

4. 卡通大小选择开关 D5 位

D5 位为定义卡通大小的选择开关。当 D5＝0 时定义 8×8 点阵的小卡通，这时卡通仅使用选定的卡通字模库的字模；当 D5＝1 时将定义 8×16 点阵的大卡通，这时，卡通将使用整个字模区(8K)。

5. 中断申请响应开关 D7 位

D7 位为中断申请响应开关。当置 1 时 CPU 响应中断申请，执行中断服务程序；置 0 时则不响应中断申请。

二、显示开关组 $2001

$2001 是屏幕显示控制开关组，各位开关的作用见图 3—9。

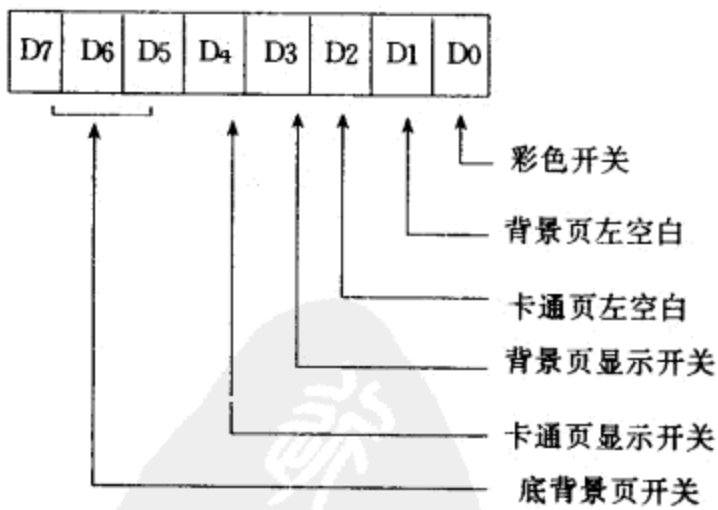


图3—9　显示开关组功能示意图

说明：

1. 彩色控制开关 D0 位

D0 位为彩色控制开关。当 D0＝1 时关闭彩色，屏幕显示呈黑白图象。实质上，D0＝1 时关闭了 PPU 内部的彩色发生

器；当 D0＝0 时则打开了彩色发生器。

2. 背景左边一列显示开关位 D1

D1 位为背景左边一列显示开关位。当 D1＝1 时，背景左边一列的内容开显示(8 点宽)；当 D1＝0 时，背景左边一列的内容不显示。

3. 卡通左边一列显示开关位 D2

D2 位为卡通左边一列显示开关位。当 D2＝1 时，卡通左边一列的内容开显示(8 点宽)；当 D2＝0 时，卡通左边一列的内容不显示。

4. 背景页显示开关 D3 位

D3 位是背景页显示开关。通常 D3＝1，仅当与 PPU 通信时，为避免屏幕闪烁，才令 D3＝0，关闭背景页显示。通信结束后再令 D3＝1，打开显示。F BASIC 中，当 D3＝0 时，将不显示光标。

5. 卡通页显示开关 D4 位

D4 位是卡通页显示开关。D4＝1 时，相当于执行 F BASIC 语句中的 SPRITE ON 指令；D4＝0 则相当于执行 SPRITE OFF。

6. 全屏幕色彩微调开关组 D5～D7 位

D5～D7 位为全屏幕色彩微调开关组。其中 D5 位决定绿色调，D5＝0 时恢复原色彩、D5＝1 时全屏幕变绿；D6 位决定黄色调，D6＝0 时恢复原色彩、D6＝1 时全屏幕变黄；D7 位决定蓝色调，D7＝0 时恢复原色彩、D7＝1 时全屏幕变蓝。有了这三个软开关，我们可在不改变背景配色的情况下，微调背景的八种显示色彩，从而使任天堂的 64 种色彩变为 512 种色彩。

第五节　CPU 与 PPU 之间的通信方法

我们知道 PPU 是图象处理器，它与 CPU 分别管理一定的内存单元。因此，它们之间的通信则是 6527 工作系统中至关重要的一环。由电路图可以看出，CPU 有三根地址线 A0～A2 与 PPU 直接相连，这就决定了 CPU 只能通过八个软开关与 PPU 打交道。这八个软开关的对应地址为 $2000～$2007，CPU 正是通过这八个软开关的状态设置来控制 PPU 的各种操作的。

以上我们曾讨论了软开关组 $2000、$2001，常用的软开关还有屏幕位移开关 $2005、设置 PPU 通信地址开关 $2006 和 PPU 读、写控制开关 $2007。下面分别介绍它们的一般使用方法。

一、读取 PPU 内存的方法

读 PPU 内存的汇编语言程序见 No. 3－3。

No. 3－3 读 PPU 内存

```
LDA  # $00
STA  $2001        ;关显示以避免屏幕闪烁
LDA  # $YY        ;YY 为欲读取的 PPU 地址高位
STA  $2006
LDA  # $XX        ;XX 为欲读取的 PPU 地址低位
STA  $2006        ;以上设置了 PPU 的读数首地址
                   $YYXX
LDA  $2007        ;触发 PPU 读写开关
LDA  $2007        ;读取 PPU 数据
```

```
PHA                 ;读出的数据入栈保存
LDA #$0E
STA $2001           ;开显示，0EH 值可根据需要改变
PLA                 ;从堆栈中弹出所读的 PPU 数据，
                     数据存在 A 寄存器中。
RTS
```

说明：

程序 No. 3－3 只是读取 PPU 内存的原理程序，实际应用时，首先要保存有关寄存器中的原值(即保护现场)；然后设置计数器，累计读数的个数；读数完毕还要恢复现场等。下面给出读取 PPU 的实用程序。

例 3－2　读出 PPU 内存中 $00～$10 单元的 17 个数据存入 $7800～$7810。

实现程序见 No. 3－4。

No. 3－4 读 PPU 内存实用程序

```
        PHA             ;保存 A 寄存器原值
        TXA
        PHA             ;保存 X 寄存器原值
        LDX #$00        ;计数器清零
        LDA #$00
        STA $2001       ;关显示
        LDA #$00
        STA $2006
        LDA #$00
        STA $2006       ;设置 PPU 首址 $0000
        LDA $2007       ;触发 PPU 读、写开关
READ    LDA $2007
```

```
STA $7800,X;读数送$7800+X单元
INX
CPX #$11    ;检查取数个数是否达到要求
BNE READ    ;没达到再读
LDA #$0E
STA $2001   ;开显示
PLA
TAX         ;恢复X寄存器原值
PLA         ;恢复A寄存器原值
RTS
```

执行 No. 3－4，即可把 PPU 的 $00～$10的数据读出存入 $7800～$7810。

程序 No. 3－4虽然已经很完美，但是在我们的学习机中还不能直接运行。下面介绍在 F BASIC 状态下使用程序 No. 3－4的方法。

1. 首先把 No. 3－4汇编成机器语言，假定我们把这段机器语言程序放在用户程序区的最后一页，从 $7F00开始存放，则 No. 3－4的机器语言程序如 No. 3－5。

No. 3－5 读 PPU 内存实用程序

```
7F00  48          PHA
7F01  8A          TXA
7F02  48          PHA
7F03  A2 00       LDX #$00
7F05  A9 00       LDA #$00
7F07  8D 01 20    STA $2001
7F0A  A9 00       LDA #$00
7F0C  8D 06 20    STA $2006
```

```
7F0F   A9 00       LDA # $00
7F11   8D 06 20    STA $2006
7F14   AD 07 20    LDA $2007
7F17   AD 07 20    LDA $2007
7F1A   9D 00 78    STA $7800,X
7F1D   E8          INX
7F1E   E0 11       CPX # $11
7F20   D0 F5       BNE $7F17
7F22   A9 0E       LDA # $0E
7F24   8D 01 20    STA $2001
7F27   68          PLA
7F28   AA          TAX
7F29   68          PLA
7F2A   60          RTS
```

2. 把程序 No. 3－5写入对应的内存单元

写入程序见 No. 3－6。

```
5  REM No. 3－6 (读 PPU 内存实用程序的写入程序)
10 DATA 72, 138, 72, 162, 0, 169, 0, 141, 1, 32, 169,
   0, 141, 6, 32, 169, 0, 141, 6, 32, 173, 7, 32, 173,
   7, 32. 157, 0, 120, 232, 224, 17, 208, 245, 169, 14,
   141, 1, 32, 104, 170, 104, 96
20 FOR I=0 TO 42: READ A: POKE &H7F00+I, A:
   NEXT
30 CALL &H7F00
40 FOR I=&H7800 TO &H7810: PRINT PEEK (I)" ";:
   NEXT
```

运行程序 No. 3－6,打印结果如下:

```
00  00  07  31  63  31  15
48  03  15  00  02  04  30
00  03  00
```

改变 No. 3－6中的 PPU 地址，可打印出其它地址的数据。

3. 对于不懂机器语言的朋友，也可以编写 F BASIC 程序读出 PPU 内存。程序见 No. 3－7。

```
5  REM No. 3－7（读 PPU 内存实用程序）
10  POKE &H2001 , 0
20  POKE &H2006 , 0
30  POKE &H2006 , A
40  B = PEEK ( &H2007 )
50  B = PEEK ( &H2007 ) : PRINT HEX$ ( B ),
60  A = A + 1 : IF A = 18 THEN 80
70  GOTO 20
80  POKE &H2001 , 14
90  END
```

运行 No. 3－7将打印出与 No. 3－6相同的结果。但是，若比较两个程序的运行时间，将会发现程序 No. 3－6 几乎是立即打印出结果，而运行 No. 3－7则要关闭显示一段时间后，才显示出运行结果。由此可见，机器语言程序的运行速度要比 BASIC 语言快许多倍。

二、写入 PPU 内存的方法

写入 PPU 内存的编程思路和方法与读 PPU 基本相同。汇编语言程序见 No. 3－8。

No. 3－8 写入 PPU 内存

```
LDA  # $00
```

```
STA  $2001        ;关显示
LDA  #$YY
STA  $2006
LDA  #$XX
STA  $2006        ;置 PPU 首址
LDA  #$MM
STA  $2007        ;把数据写入 PPU
LDA  #$0E
STA  $2001        ;开显示
RTS
```

一旦设置好 PPU 的起始地址后，可连续写入多个数据。关于机器语言的写入程序，读者可参照程序 No. 3－6、No. 3－7 自己编写。下面仅给出使用 F BASIC 语言编程写入 PPU 内存的方法。

例3－3　从背景0页的第一行起，显示背景图形库(第二区图形)的256个字符、图形。

实现程序见 No. 3－9。

```
5  REM No. 3－9 (写 PPU 内存)
10  POKE 8193 , 0
20  POKE 8192 , 148
30  FOR I = 0 TO 255
40  POKE 8198 , 32
50  POKE 8198 , I
60  POKE 8199 , I
70  NEXT
80  POKE 8193 , 14
90  END
```

运行程序，将打印出32列×8行的256个字符和图形。

程序 No. 3－9 提供在 F BASIC 状态下设计32列×30行游戏画面的手段。

第六节　屏幕显示窗口

一、显示窗口的概念

屏幕显示窗口是一个形象的称谓。实际上，它是由卡通0页、卡通1页、底背景页三层画面叠加而成的一个具有立体结构的画面。因此，显示窗口的平面尺寸为32列×30行。为方便讨论，我们把它抽象为一个具有32列×30行的透明矩形，在示意图中以粗线框表示，讨论中简称为显示窗口或直接称窗口。

虽然画面位移的实质是显示窗口移动，但为了符合画面位移的通常概念和称谓习惯，我们在讨论中仍称背景画面位移，但在理论上要清楚实际是显示窗口在移动。

二、显示窗口的坐标系

为使显示窗口的坐标与卡通显示坐标相对应，我们把窗口的坐标原点选在左上角（如图3－10所示），坐标值按点阵计算，则 XMAX＝256、YMAX＝240。实用中 X 取值0～255、Y 取值0～239。

三、显示窗口与背景画面的对应关系

由于画面的移动，实质上是窗口向相反方向的位移，那么对于图3－10选定的窗口坐标系，背景画面与显示窗口将

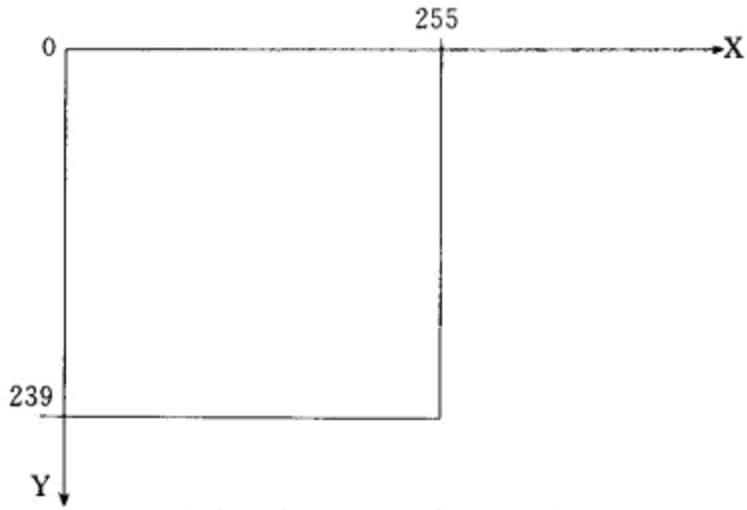


图3—10　显示窗口坐标系

有图3—11所示的对应关系。

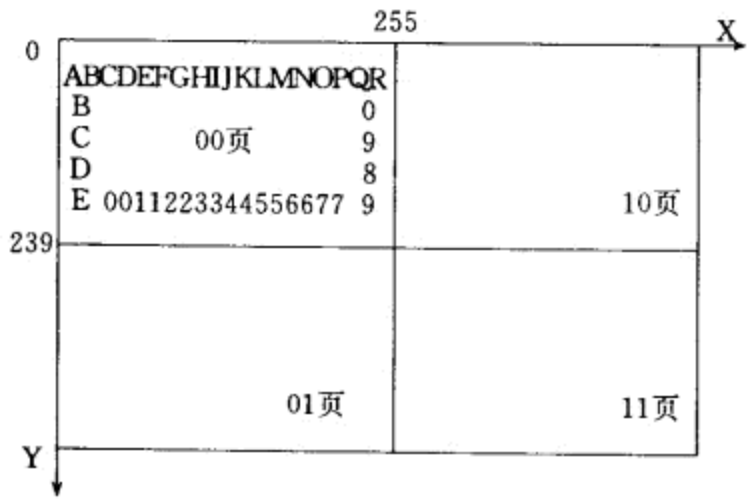


图3—11　显示窗口与背景画面的关系图

第七节　显示系统的配色

一、6527的配色区

我们知道，在F BASIC状态下，当使用CGSET指令选定背景和卡通配色板后，就可以在DEF MOVE、DEF

SPRITE 等指令中使用选定配色板中的配色代码。由于每个配色板有12种颜色，故两个配色板有24种颜色供选用。当我们欲自己组合颜色时，可使用 PALETB、PALETS 指令进行定义，一经定义后就可以在显示画面中使用。另外，我们还可以在背景配色板或卡通配色板的0＃配色代码的重新设定中改变底背景的配色，使用极为方便。但是，我们自己设定的配色数据存于系统 RAM 区的什么位置呢？查遍 RAM 区也找不到。原来6527CPU 工作系统有一专用的配色区，这一配色区位于 PPU 内部，寻址地址是＄3F00～＄3FFF。

二、6527配色区的地址分配

在 PPU 的＄3F00～＄3FFF 这256个单元中，存储了各种颜色代码，其地址分配见表3－6。

表3－6　PPU 配色地址分配表

地　　址	用　　途
3F00	底背景配色
3F01～3F03	背景配色板0＃配色代码的颜色代码
3F05～3F07	背景配色板1＃配色代码的颜色代码
3F09～3F0B	背景配色板2＃配色代码的颜色代码
3F0D～3F0F	背景配色板3＃配色代码的颜色代码
3F10	底背景配色
3F11～3F13	卡通配色板0＃配色代码的颜色代码
3F15～3F17	卡通配色板1＃配色代码的颜色代码

续表

地　　址	用　　途
3F19～3F1B	卡通配色板2＃配色代码的颜色代码
3F1D～3F1F	卡通配色板3＃配色代码的颜色代码
＊3F20～3FEF	映射于3F00～3F0F
＊3F30～3FFF	映射于3F10～3F1F

注——①在＄3F20～＄3FEF 中凡以＄3F20、＄3F40、＄3F60等开始的连续的16个单元均映射于＄3F00～＄3F0F。

②在＄3F30～＄3FFF 中凡以＄3F30、＄3F50、＄3F70等开始的 连续的16个单元均映射于＄3F10～＄3F1F。

③凡＄3F04、＄3F08、＄3F0C、＄3F14、＄3F18、＄3F1C 等最 末一位是4、8、CH 的单元没有使用，一般置入0FH。

例3－4　写出在F BASIC 状态下执行“CGSE T 1，0”和“CGSET 0，2”命令后各配色单元的内容。

执行“CGSET 1，0”后，使用读 PPU 内存的程序可读出各配色单元内容如下：

3F00　0F 30 21 02 0F 30 27 18

3F08　0F 30 27 16 0F 29 36 17

3F10　0F 36 16 02 0F 27 30 19

3F18　0F 35 25 17 0F 30 27 16

执行“CGSET 0，2”后，各配色单元的内容如下：

3F00　0F 2C 15 07 0F 27 21 12

3F08　0F 29 36 17 0F 30 26 07

3F10　0F 30 26 12 0F 30 15 12

3F18　0F 30 12 16 0F 30 26 19

当我们执行“PALETB 0，3，5，7，9”时则底背景变为蓝色，相应的配色单元变为

3F00　03 05 07 09 0F 27 21 12

读者可使用读 PPU 内存的方法查看、验证各配色单元的内容；也可以用直接写入 PPU 的方法，不使用 PALET 指令而改变底背景或背景、卡通的配色。

三、说明

1. 在配色区，每四个单元为一组，每组的后三个单元存颜色代码。

2. 在各个配色单元中，只有 PPU 地址的末位为0的单元为底背景配色单元，其它位无效。

3. 如果把 $3F00～$3FFF 分成16行，那么偶数行之间互相等效（包括0行，即0、2、4、6……），也就是说对 $3F60～$3F6F 操作等效于对 $3F00～$3F0F 操作，都作用于背景页；同样奇数行之间互相等效（1、3、5、7……），它们都作用于卡通页。

第八节　6527CPU 的中断系统

我们知道，6527CPU 最常用的中断方式是非屏蔽中断 NMI。

非屏蔽中断一般是指不能通过编程进行阻止的中断。只要非屏蔽中断请求信号一发出，中央处理器在执行完当前指令后就立即响应。但由 $2000 软开关组的 D7 位可知，6527CPU 的非屏蔽中断实际上是可屏蔽的，只要 D7＝0就可以控制 CPU 不执行中断服务程序。这也是6527CPU 的一个

特点。

一、NMI 中断的执行过程

6527CPU 的中断入口地址放在 $FFFA(地址低位)和 $FFFB(地址高位)中，F BASIC 中一般指向 $00ED。

当 CPU 响应中断时，则自动保存断点地址和状态，即将当前的程序计数器 PC 值和状态寄存器 P 值送入堆栈保存；然后根据中断矢量单元 $FFFA、$FFFB 提供的中断程序入口地址跳转中断服务程序进行中断处理；当中断处理结束后，则返回到断点地址处，恢复原来的内部状态，继续执行原程序。

二、NMI 中断的作用

NMI 中断在6527CPU 的工作系统中起着巨大的作用。任天堂游戏中的音响发出、画面更换、角色控制等操作都是在中断程序中完成的；人机对话工作系统的全部操作，几乎都是在中断中进行的；F BASIC 工作系统中，动作卡通的运动也是在中断中实现的。因此，NMI 中断在6527CPU 工作系统中占有举足轻重的地位。甚至可以这样说：没有 NMI 中断就没有任天堂游戏。强大的中断功能是6527CPU 的一个重要特点。

三、中断程序的设计

建立一个性能良好的中断环境是编好6527软件的关键。中断程序的设计一般采取如下的方法：

(1)从0页选定3个单元 P0、P1、P2，其中 P1、P2两单元地址相连。

(2)编写中断程序

中断程序一般采取如下的模式，见程序 No. 3－34。

No. 3－34 中断程序的一般模式

```
    PHA
    LDA  $P0              ;取系列中断程序入口地址偏移量
    BNE SA                ;不等于0则转SA 进行中断处理
    PLA
    RTS                   ;等于0则中断返回

SA  TAX
    DEX
    TXA
    ASL
    TAX
    LDA SUBDATA , X       ;取系列中断程序的地址低位
    STA  $P1
    LDA SUBDATA+1 , X     ;取系列中断程序的地址高位
    STA  $P2
    JMP ( $P2P1 )         ;转入中断处理程序
```

注——SUBDATA 为系列中断程序入口地址的数据地址，系列中断程序可分为：读 PPU 内存、写 PPU 内存、音响发声控制等操作。至于动画处理、读操纵器状态等操作也可以安装在此处。具体程序将在以后章节的分析中看到。

另外，在中断处理程序中，一开始还必须保存 A、X、Y 寄存器的数值，返回之前恢复。

若中断处理程序中不将 P0清零，则调用一直持续，直至清零为止。显然，通过对 P0赋值，可以控制转移到不同的中断处理程序进行不同的控制操作。

四、中断状态的进入

下面通过一个实例，介绍中断状态的进入方法。

例3－7　设计一计时程序，用于检测程序的运行时间。

汇编程序见 No. 3－35(假定我们把机器语言程序安装在 $7F00开始的 RAM 中)。

No. 3－35 计时程序

```
7F00   E6 D2       INC  $D2
7F02   D0 02       BNE  $7F06
7F04   E6 D3       INC  $D3
7F06   4C 0F 88    JMP  $880F
7F09   A9 00       LDA  #$00
7F0B   85 EE       STA  $EE
7F0D   A9 7F       LDA  #$7F
7F0F   85 EF       STA  $EF
7F11   60          RTS
```

说明：程序 No. 3－35适用于中断程序入口为 $880F 的版本。读者可使用

```
PRINT HEX$ ( PEEK ( &HEE ) )
PRINT HEX$ ( PEEK ( &HEF ) )
```

检查。若打印结果为0FH、88H，则可直接使用 No. 3－35；若是另外的数值，则把第一个数据置入 $7F07、第二个数据置入 $7F08，即可使用程序 No. 3－35。

调用 No. 3－35的 F BASIC 程序见 No. 3－36。

```
5   REM No. 3－36
10  FOR I=&H7F00 TO &H7F11: READ A: POKE I,A:
    NEXT
```

```
20  DATA 230, 210, 208, 2, 230, 211, 76, 15, 136, 169,
    0, 133, 238, 169, 127, 133, 239, 96
30  END
65510    POKE &HD3, 0: POKE &HD2, 0: CALL
         &H7F09: RETURN
65520  END
65530  A=PEEK(&HD3): B=PEEK(&HD2): C=A*5
       +(A*6+B)/50: PRINT " TIME= ";C: RE-
       TURN
```

程序 No. 3－36的使用说明：

(1)程序 No. 3－36共分三段：

10～30行为输入 No. 3－35程序的处理程序，该段程序运行后可删去而输入其它应用程序；

65510行是计数器清零处理，计时开始时要先对计数器清零；

65530行为计时结果输出程序。

(2)当需要计时时，首先调用65510子程序，对计时单元$D2、$D3清零，以启动计时。

(3)当需要检查计时时间时，只要调用65530子程序即可打印出计时的时间，时间单位为秒。

第四章　背景画面的位移

背景画面位移一般分为横向位移、纵向位移和斜向位移三大类。通常称横向位移为画面的左右卷动、纵向位移为画面的上下滚动。多数 CPU 的位移操作是通过对屏幕映射单元不停地改变数据实现的。而6527CPU 的位移方法则另辟蹊径，采用了屏幕显示映射单元的内容不变，通过移动屏幕显示窗口实现画面位移的方法。这犹如照相机或摄像机中的取景框，显示窗口对准了哪一块图形，就把哪一块图形显示在屏幕上。因此，背景画面的位移实质就是显示窗口向相反的方向移动。由于显示窗口可任意移动，因此6527CPU 的背景画面可方便地进行任意方向的移动（一般取八个方向）。由于每一个方向的位移可分为往复、循环两种过程，所以6527CPU 一般有16种位移方式。

繁多的位移方式和简单的位移操作是6527CPU 的突出特点。

下面分别介绍各类位移操作的原理和实现方法。

第一节　画面的简单位移

一、画面的横向位移

假定画面位移前，背景00页的内容如图4－1所示。

画面横向位移的示范程序如 No. 4－1。

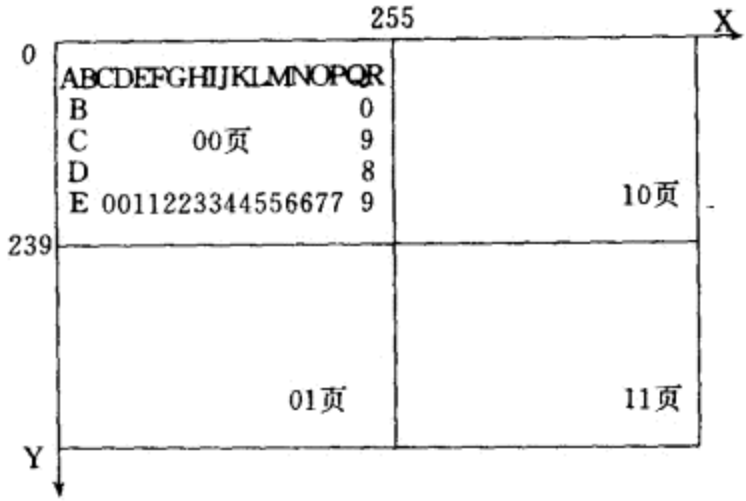


图4—1　画面位移前的状态

（以移动 a 点为例）

No. 4—1　　画面横向位移 a 点

```
LDA  # $a
STA  $2005      ;窗口横向位移
LDA  # $00
STA  $2005      ;窗口纵向位移
JSR YAN SHI     ;延时
RTS
```

执行程序 No. 4—1，将有图4—2所示的画面。

程序 No. 4—1的 F BASIC 程序如 No. 4—2所示。

```
5 REM No. 4—2（画面横向位移 a 点）
10 POKE 8197 , A
20 POKE 8197 , 0
30 PAUSE 1000 : END
```

二、画面的纵向位移

若令画面纵向位移 b 点，其实现程序如 No. 4—3。

No. 4—3 画面纵向位移 b 点

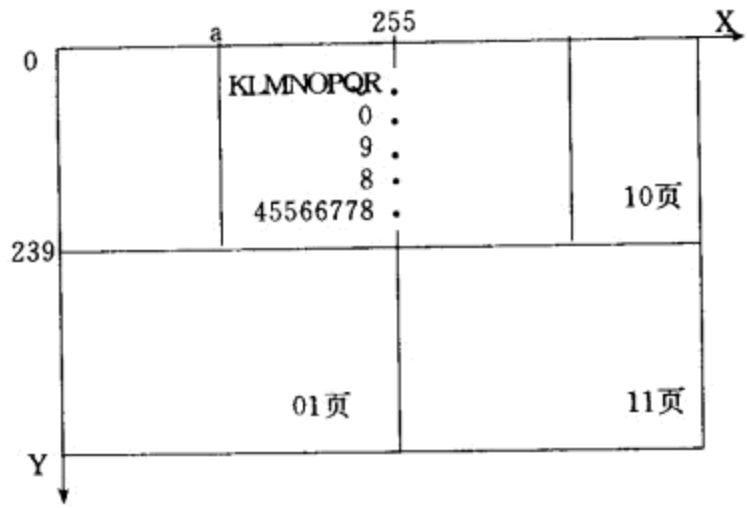


图4—2　画面横向移动a点示意图

```
LDA  # $ 00
STA  $ 2005
LDA  # $ b
STA  $ 2005
JSR YAN SHI
RTS
```

执行程序 No. 4—3，将有图4—3所示的画面。

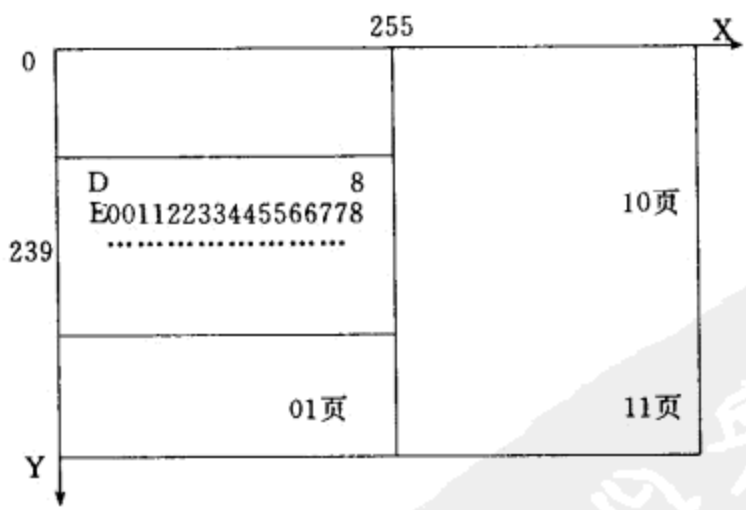


图4—3　画面纵向移动b点的示意图

三、画面的斜向位移

若令画面横向位移 c 点、纵向位移 d 点，其实现程序如 No. 4－4。

No. 4－4 画面斜向位移

```
LDA  # $c
STA  $2005
LDA  # $d
STA  $2005
JSR YAN SHI
RTS
```

执行程序 No. 4－4，将有图4－4所示的画面。

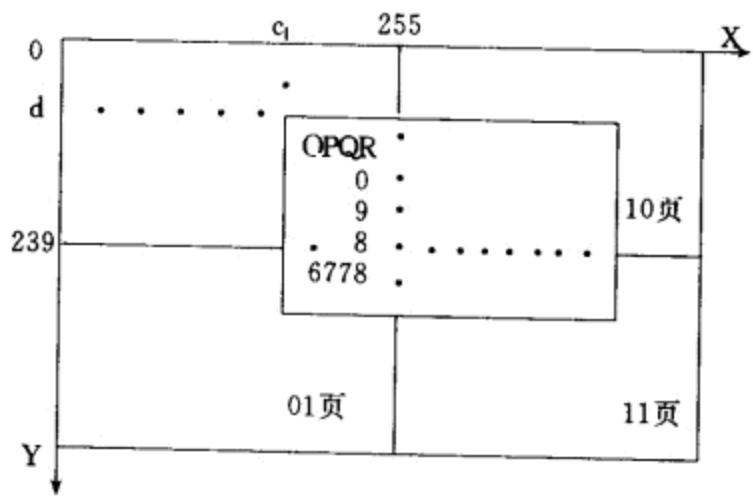


图4—4　画面斜向位移示意图

通过以上的位移过程可见，6527CPU 的画面移动操作是极为简便的。

第二节　画面的连续位移

实际游戏中，画面的位移操作多数是以点为单位的连续位移。如《碰碰车》游戏是连续的横向位移；《拆屋工》是连续的纵向位移；《淘金者》、《沙罗曼蛇》等游戏则是横向、纵向连续分段位移；而大多数游戏如《魂斗罗》、《松鼠大作战》、《上尉密令》、《人间兵器》等游戏则是横向、纵向混合位移。从而使游戏画面广阔、博大、丰富迷人。

一般把画面的横向连续位移称为横向卷动，把画面的纵向连续位移称为纵向滚动。下面分别讨论各类连续位移的实现方法。

一、画面连续左移

从图4－2可以看出，画面的横向位移是在00、10页或01、11显示页面间的移动。根据程序 No. 4－1，我们很容易写出画面连续左移的程序 No. 4－5。

No. 4－5 画面连续左移

```
      LDX  # $00
      LDY  # $00
LEFT  STX  $2005
      STY  $2005
      JSR YAN SHI
      INX
      BNE LEFT
      RTS
```

对应的 F BASIC 程序见 No. 4－6。

```
5  REM No. 4－6（画面连续左移）
10 A = 0
20 POKE 8197 , A
30 POKE 8197 , 0
40 A = A + 1 : IF A = 256 THEN PAUSE 1000 : END
50 PAUSE 5
60 GOTO 20
```

执行程序 No. 4－5或 No. 4－6 可把画面的00页连续左移出窗口，使窗口对正10页。

调整50行 PAUSE 语句的参数，可控制画面左移的速度。

二、画面的连续右移

画面的连续右移程序见 No. 4－7。

No. 4－7 画面连续右移

```
        LDX # $FF
        LDY # $00
RIGHT   STX $2005
        STY $2005
        JSR YAN SHI
        DEX
        BNE RIGHT
        RTS
```

对应的 F BASIC 程序见 No. 4－8。

```
5  REM No. 4－8（画面连续右移）
10 A = 255
20 POKE 8197 , A
30 POKE 8197 , 0
```

```
40  A = A - 1 : IF A = -1 THEN END
50  PAUSE 5
60  GOTO 20
```

画面连续右移时，首先把窗口迅速移到10页面，然后连续右移，直到把00页全部移入显示窗口而停止。

三、画面的连续上移

由图4－3可见，画面的纵向位移是在00、01或10、11页面间移动。画面连续上移的程序见 No. 4－9。

No. 4－9 画面连续上移

```
    LDX  # $00
    LDY  # $00
UP  STX  $2005
    STY  $2005
    JSR YAN SHI
    INY
    CPY  # $F0
    BNE UP
    RTS
```

对应的 F BASIC 程序见 No. 4－10。

```
5   REM No. 4－10 (画面连续上移)
10  A = 0
20  POKE 8197 , 0
30  POKE 8197 , A
40  A = A + 1 : IF A = 240 THEN PAUSE 1000 : END
50  PAUSE 2
60  GOTO 20
```

四、画面的连续下移

画面的连续下移程序见 No. 4－11。

No. 4－11 画面连续下移

```
        LDX  # $00
        LDY  # $F0
DOWN    STX  $2005
        STY  $2005
        JSR YAN SHI
        DEY
        BNE DOWN
        RTS
```

对应的 F BASIC 程序见 No. 4－12。

```
5   REM No. 4－12 (画面连续下移)
10  A = 240
20  POKE 8197 , 0
30  POKE 8197 , A
40  A = A － 1 : IF A = －1 THEN END
50  PAUSE 2
60  GOTO 20
```

同画面的连续右移的情况相同，下移前显示窗口要迅速跳到01页，然后逐点上移到00页，直到窗口与00页对正后，位移过程结束。

第三节　画面的往复位移

一、画面的横向往复位移

《淘金者》二代游戏中，按暂停键后，可用左右方向键控制画面左右往复运动。这种位移过程的控制程序实际上是把程序 No. 4－5、No. 4－7合并组成的。合并后的程序见 No. 4－13。

No. 4－13 画面的横向往复位移

```
        LDX # $00
        LDY # $00
LEFT    STX $2005
        STY $2005
        JSR YAN SHI
        INX
        BNE LEFT
        DEX
RIGHT   STX $2005
        STY $2005
        JSR YAN SHI
        DEX
        BNE RIGHT
        BEQ LEFT
```

对应的 F BASIC 程序为 No. 4－14。

```
5   REM No. 4－14(画面的横向往复位移)
10  A = 0
```

```
20  POKE 8197 , A
30  POKE 8197 , 0
40  IF B = 0 THEN A = A + 1 : IF A = 256 THEN B =
    1
50  IF B = 1 THEN A = A - 1 : IF A = 0 THEN B = 0
60  PAUSE 5
70  GOTO 20
```

运行程序 No. 4－13或 No. 4－14 就可看到画面先连续左移到背景10页；然后又连续右移到背景00页，不停地往复移动，直到按 STOP 键才停止位移。

为使画面的位移过程更易分辨，我们可以在执行 No. 4－13或 No. 4－14之前，把背景00、10两页都写入内容。背景0页常用的写入方法是在程序中加一条输出语句，向00页写入字符或图形。背景1页的写入方法有两种：一种是直接把字符数据写入 PPU；第二种方法是在 BG GRAPHIC 画面绘制图案或字符(因为 BG GRAPHIC 画面使用背景10页面)。

10页面最简单的绘图方法就是使用 BG GRAPHIC 画面键控绘图。操作过程是：在 BS·2AF BASIC 状态下，键入 SYSTEM 命令，即返回选项画面；选择2，进入 BG GRAPHIC 绘图状态，进行键控绘图；绘图结束后，返回 F BASIC 状态，在程序 No. 4－14中加入

```
15 CLS : FOR I=0 TO 600 : PRINT " A " ; : NEXT
```

再运行程序，就可以直观、清晰地观察画面的往复位移过程。

另外，在汇编语言位移程序中，我们调用了一个延时子程序“YAN SHI”，可能朋友们编写的延时子程序延时量太小或延时程序较大，使用起来不太方便。现给出一个延时量

任意可调的通用延时子程序，见程序 No. 4－15。

No. 4－15 YAN SHI(延时)

```
YAN  SHI LDA # $a
         STA  $401
         LDA  # $b
         STA  $400
         JSR  nn
         RTS
```

程序中 a、b 为延时参数。其中 b 为延时数据低位，b 的取值延时量与 PAUSE 语句中的参数相同；a 为延时数据高位，a 值的一个单位相当于 PAUSE256。故一般取 a＝0、b＝2～5。“nn”为系统延时子程序的入口地址，对于 BS. 2A 版本的 F BASIC nn＝$91B0；对于 V3·10版本的 F BASIC nn＝$9731。使用“YAN SHI”子程序时，只要根据程序延时的要求，用相应的数值代入 a、b 中就可任意调整延时时间。

二、画面的纵向往复位移

由于画面的纵向往复位移仅在背景0页或1页的本页之间进行，所以其实现程序更为简单。借鉴横向往复位移的编程方法，将很容易写出画面纵向往复位移的实现程序，见 No. 4－16。

No. 4－16 画面的纵向往复位移

```
         LDX  # $00
         LDY  # $00
UP       STX  $2005
         STY  $2005
         JSR  YAN SHI
```

```
        INY
        CPY # $F0
        BNE UP
DOWN    STX $2005
        STY $2005
        JSR YAN SHI
        DEY
        BNE DOWN
        BEQ UP
```

对应的 F BASIC 程序为 No. 4－17。

```
5  REM No. 4－17（画面的纵向往复位移）
10 A = 0
15 CLS : FOR I = 0 TO 600 : PRINT " A " ; : NEXT
20 POKE 8197 , 0
30 POKE 8197 , A
40 IF B = 0 THEN A = A + 1 : IF A = 240 THEN B =
   1
50 IF B = 1 THEN A = A － 1 : IF A = 0 THEN B = 0
60 PAUSE 5
70 GOTO 20
```

第四节　画面的循环位移

画面的循环位移，包括画面的横向连续卷动、纵向连续滚动和斜向连续卷动三类位移。

一、画面的纵向连续滚动

所谓画面的纵向连续滚动就是画面的头尾相接，纵向旋转。画面纵向连滚的程序见程序 No. 4－18～No. 4－21。

1. 画面向上连续滚动

No. 4－18（画面向上连续滚动）

```
    LDX # $00
    LDY # $00
UP  STX $2005
    STY $2005
    JSR YAN SHI
    INY
    CPY # $F0
    BNE UP
    WLDY # $00
    BEQ UP
```

对应的 F BASIC 程序如下：

```
5   REM No. 4－19（画面向上连续滚动）
10  A = 0
20  POKE 8197 , 0
30  POKE 8197 , A
40  A = A + 1 : IF A = 240 THEN A = 0
50  PAUSE 2
60  GOTO 20
```

2. 画面向下连续滚动

No. 4－20（画面向下连续滚动）

```
        LDX # $00
```

```
        LDY # $EF
DOWN    STX $2005
        STY $2005
        JSR YAN SHI
        DEY
        BNE DOWN
        LDY # $EF
        BNE DOWN
```

对应的 F BASIC 程序为：

```
5   REM No. 4－21（画面向下连续滚动）
10  A = 239
20  POKE 8197 , 0
30  POKE 8197 , A
40  A = A － 1 : IF A = －1 THEN A = 239
50  PAUSE 2
60  GOTO 20
```

3. 画面纵向连滚的工作原理

从程序 No. 4－18～No. 4－19的运行中可以明显看出，在画面的纵向连续滚动过程中，画面没有间断、没有跳跃，一直连续、匀速、圆滑地位移。这种现象，用图4－3、图4－4的理论显然解释不通。那么，这种位移的工作原理是怎样的呢？

原来6527CPU 的显示页面并不象图4－1所示的那样简单地排列，实际上要复杂的多。为便于理解，我们采取分步讨论、循序过渡的方式进行解释。

几点假设：

(1)把背景页假设为形如大口径水泥管子状的一个横向放置的圆筒，不妨称它为背景筒。显然，背景筒是由四个显示

页面横向弯曲变形、上下两边对接而成的。

(2)假设在背景筒外面又套上了一个稍大于背景筒的圆筒，不妨称它为显示筒。显示筒由不透光的材料制成，在显示筒上开有一个32列×30行或256×240点的矩形洞，这个矩形洞就是屏幕显示窗口。显然，现在的显示窗口已是一个曲面窗口。

(3)假设在显示窗口的四个角上，向窗口的重心方向伸出四根支柱，以这四根支柱做支撑，在柱子的顶端搭一个平台，我们就站在这个平台上观察窗口。假定柱子足够长，我们离开显示窗口的距离足够远，则在我们的视觉中弯曲的窗口呈现出平面的图象。

以上三点假设如图4－5所示：

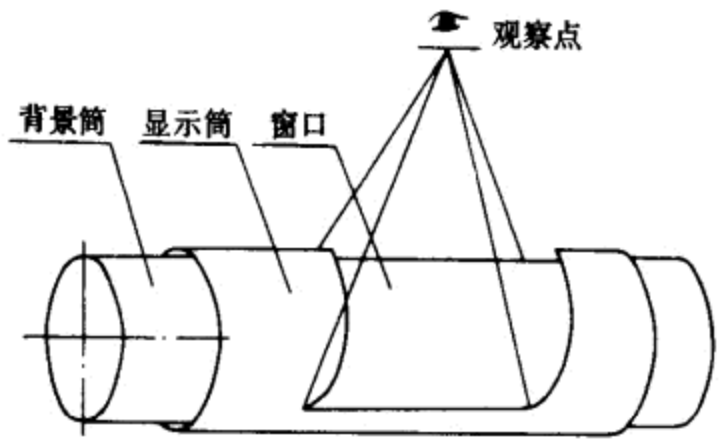


图4－5　画面纵向连滚原理示意图

(4)假设显示筒由一同步电机带动，可绕背景筒正、反旋转；同时，还与一步进电机相连接，可带动显示筒沿背景筒左右滑动。

(5)假设我们站在观察点上看屏幕，观察点随显示筒一起运动，观察者与显示窗口之间保持同步。

从以上五点假设出发，可很容易解释画面上下连滚时的

工作原理。

·当画面在背景0页向上连滚时，则由程序 No. 4－19控制窗口在00、01两个显示页面连成的圆筒上面向下旋转。由于00、01两页显示同一幅画面，因此，在连续上滚时，看到的画面是上一行内容刚移出屏幕又从最底行显示出来。从视觉上似乎背景筒由一个页面组成，实际上，移出的是00页的第一行，下面显示的是01页的第一行。

·当画面在背景1页向上连滚时，则对 $2000单元的 D0位置1，相当于接通步进电机的电源，由步进电机把显示窗口推向1页背景筒部位。这时，窗口在由10、11显示页组成的背景筒上旋转，旋转显示过程同0页。

·画面向下连滚的情况与上滚基本相同，差别仅是令同步电机反转而已。

二、画面的横向连续卷动

画面的横向连续卷动就是00、10或01、11两幅显示页面头尾相连、连续左移或连续右移的过程。实现程序如下：

1. 画面的连续左卷

```
        No. 4－22（画面连续左卷）
        LDA  # $90
        STA  $2000
        STA  $FF
        LDX  # $00
        LDY  # $00
LEFT    STX  $2005
        STY  $2005
        JSR YAN SHI
```

```
        INX
        BNE LEFT
        LDA $FF
        EOR #$01
        STA $FF
        STA $2000
        LDX #$00
        BEQ LEFT
```

对应的 F BASIC 程序如下：

```
5   REM No. 4－23（画面连续左卷）
10  A = 0
20  POKE 8197 , A
30  POKE 8197 , 0
40  A = A + 1 : IF A = 256 THEN 70
50  PAUSE 5
60  GOTO 20
70  IF B = 0 THEN POKE 8192 , 145 : B = 1 : GOTO 10
80  IF B = 1 THEN POKE 8192 , 144 : B = 0 : GOTO 10
90  END
```

运行程序 No. 4－22、No. 4－23以前，要先对背景0页、1页置入显示内容，以利观察。

2. 画面的连续右卷

画面的连续右卷是左卷的逆操作，只要修改 No. 4－23的10、40行语句就可实现。下面仅给出右卷的 BASIC 程序，见程序 No. 4－24。

```
5   REM No. 4－24（画面连续右卷）
10  A = 255
```

```
20  POKE 8197 , A
30  POKE 8197 , 0
40  A = A - 1 : IF A < 0 THEN 70
50  PAUSE 5
60  GOTO 20
70  IF B = 0 THEN POKE 8192 , 145 : B = 1 : GOTO 10
80  IF B = 1 THEN POKE 8192 , 144 : B = 0 : GOTO 10
90  END
```

3. 画面横向连续卷动的工作原理

显然，图4－2、图4－4都不能解释画面横向连卷的机理。我们可以采取画面纵向连滚的假设思路，把画面设想为一个竖立的圆筒，相应的显示窗口也随之变动，示意见图4－6。

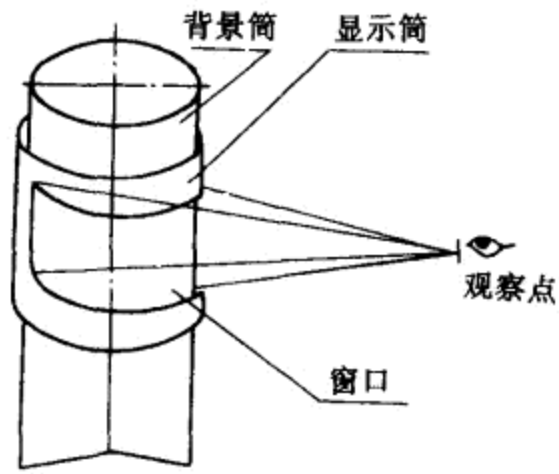


图4－6　画面横向连卷原理示意图

图4－6中，显示页面纵向弯曲，背景00、10和01、11页面的始端、末端对应相接，连成一竖立的圆筒，从而较好地解释了横向连卷的机理（具体过程留给读者分析。）

三、画面的斜向连续卷动的工作原理

把画面的纵向、横向连续位移程序合理搭配组合就可以方便地写出背景画面四个斜向连续卷动的程序。这一工作留给读者练习。下面重点讨论画面斜向连续卷动的工作机理。

当画面斜向连卷时，既有横向连卷的操作又有纵向连滚的过程。因此，图4－5、4－6都不能解释这一工作机理。那么，画面斜向卷动的工作机理应如何解释呢？

解决的方法是：对图4－5、4－6显示画面的变形过程综合考虑，把四页显示画面连接成为一个管状圆环。即把图4－5中的背景筒弯曲变形，两端口对接成环状，可称为背景环；相应地显示筒也弯曲变形为套在背景环上的一截圆环；显示窗口和观察点的设定仍如图4－5，这样就构成了图4－7的示意图。

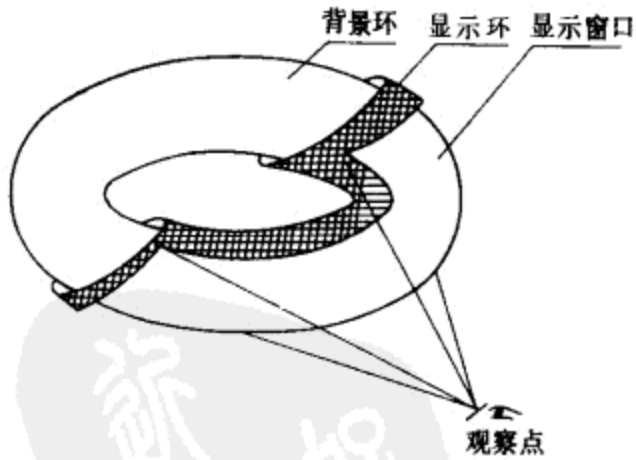


图4－7　画面位移原理示意图

在斜向连移过程中，显示环一边沿背景环滑动、一边绕背景环自身旋转，从而使四个显示页面的任一部分组合都能

在窗口显示。

注——实物中，硬质的大环是不能绕小环旋转的。为了方便理解画面斜向连卷的工作机理，我们假定：显示环是由一种柔软的、富有弹性的材料制成，它既可沿背景环滑动又可绕背景环转动，且无论怎样运动，显示窗口都能与显示页面的形状相吻合。

根据以上假设，从图4－7出发，背景画面的各种位移工作原理都可得到圆满的解释：

·画面的横向往复位移过程是显示筒在背景筒上往复滑动；

·画面的横向连续位移是显示筒沿背景筒朝一个方向连续滑动；

·画面的纵向往复位移是显示筒绕背景筒往复旋转；

·画面的纵向连续位移是显示筒绕背景筒沿一个方向连续旋转；

·画面的斜向往复位移是显示筒一边绕背景筒往复旋转，一边沿背景筒往复滑动；

·画面的斜向连续位移是显示筒一边绕背景筒向一确定转向旋转，一边沿背景筒向一确定方向滑动。

第五章　图形字模设计

顾名思义，“字模”就是字的模样。

无论 BASIC 语言、任天堂游戏，都涉及到字符或图形的显示问题。这些显示的字符或图形的模样是怎样确定的呢？这就是本章讨论的重点——图形字模设计。

在弄清图形字模的概念、结构、设计方法的基础上，将在电脑学习机显示方式的讨论中，向大家披露电脑学习机的中文显示方式、显示原理和显示操作。

第一节　字模与图形库

在 F BASIC 编程中，字符或图形的显示，一般使用两种方法：一种方法是直接把字符或图形写在 PRINT 语句的输出项中或字符变量中；另一种方法是把字符的 ASCII 码或图形对应的内码写入程序中。对于字符来说，前一种方法使用的多些；对于图形则是后一种方法经常使用。不管采用那种方法，系统存储区都将存入该字符或图形对应的内部代码，即背景图形代码表中的代码。对于显示卡通的定义也是如此，CHR $ 语句中的数字即为卡通的图形代码。

为什么在程序中写入图形的代码就可以显示出对应的图形呢？这要从图形的点阵谈起。

一、图形的点阵

在任天堂游戏中，无论背景、卡通都采用了对图形代码操作的显示方法。采用这种方法的优点是：画面处理速度快、占用显示内存少。

由于图形代码用一个字节的十六进制数据表示，所以由它限定了图形代码的个数最多为256个，编号为0～255(0～FFH)。其中每一个代码对应一个8×8点阵的图形块，也就是说每个图形块由64个点的方阵组成。方阵中的点与电视机屏幕上的显示点相对应，对于确定的图形块，哪个点亮、哪个点灭也必然确定。因此控制点的亮、灭就控制了图形的形状。如阿拉伯数字"1"的各点亮、灭排列如图5－1所示。

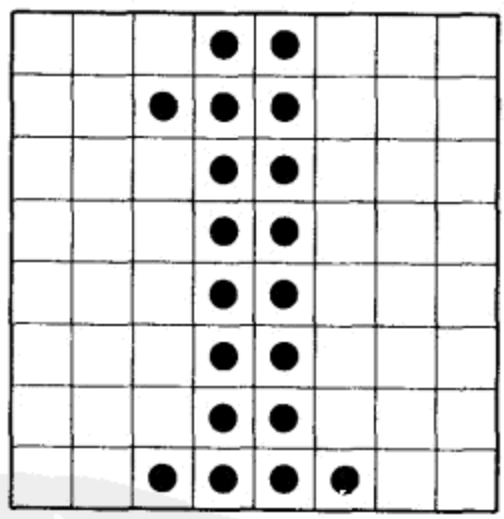


图5－1 "1"的各点亮、灭示意图

二、字模的概念

如果用"1"代表某点亮、用"0"代表某点灭，图5－1将有图5－2的形式。

0	0	0	1	1	0	0	0	18
0	0	1	1	1	0	0	0	38
0	0	0	1	1	0	0	0	18
0	0	0	1	1	0	0	0	18
0	0	0	1	1	0	0	0	18
0	0	0	1	1	0	0	0	18
0	0	0	1	1	0	0	0	18
0	0	0	1	1	0	0	0	18
0	0	1	1	1	1	0	0	3C

图 5－2 “1”的各点代码排列示意图

如果把图 5－2 中的每一行看作 8 位二进制数，8 行将有 8 个数据，写为十六进制的形式就是图 5－2 右侧的数据。即有

18　38　18　18　18　18　18　3C

以上 8 个数据就唯一确定了“1”的形状。这 8 个数据就称为字符“1”的“字模”数据，简称字模。

三、任天堂游戏中的字模形式

图 5－2 中，每个点对应一位二进制数，只能表示该点的一个状态(亮或灭)，所以由它们构成的字模数据一般称为单色字模。

在任天堂游戏中，显示的是彩色图案，每一个点可能取不同的颜色。因此任天堂游戏中的每个显示点使用两位二进制数表示。因两位二进制数可以表示 4 种状态，所以任天堂游戏中的每个点可以使用 3 种颜色点亮(一种状态表示灭)。

因一个点使用两位二进制数，64 点使用 128 位，占 16 个字节。所以任天堂游戏中每个图形块的字模数据为 16 个字

节。对于字符“1”，若选择“00”号颜色表示点“灭”、“01”号颜色表示点“亮”，其字模数据结构形式如图 5－3 所示。

列：	1	2	3	4	5	6	7	8	
行:00	0	0	0	0	0	0	0	0	
一	0	0	0	1	1	0	0	0	18
二 00	0	0	0	0	0	0	0	0	
	0	0	1	1	1	0	0	0	38
三 00	0	0	0	0	0	0	0	0	
	0	0	0	1	1	0	0	0	18
四 00	0	0	0	0	0	0	0	0	
	0	0	0	1	1	0	0	0	18
五 00	0	0	0	0	0	0	0	0	
	0	0	0	1	1	0	0	0	18
六 00	0	0	0	0	0	0	0	0	
	0	0	0	1	1	0	0	0	18
七 00	0	0	0	0	0	0	0	0	
	0	0	0	1	1	0	0	0	18
八 00	0	0	0	0	0	0	0	0	
	0	0	0	1	1	0	0	0	3C

图 5－3　任天堂游戏中“1”的字模数据结构

图 5－3 中，每个点的两位二进制代码竖写，高位在上、低位在下。每一行高位的 8 位二进制数写在左侧、低位的 8 位二进制数写在右侧。把高位的 8 个字节的数据排在前面，低位的 8 个字节数据排在后面，就得到了任天堂游戏中一个图块的字模数据。显然“1”的字模数据为：

00 00 00 00 00 00 00 00 18 38 18 18 18 18 18 3C

若字符“1”选择“10”号颜色为背景，其字模结构将如图 5－4 所示。

显然，改变了背景颜色后，字模数据将发生改变，图 5－4 的字模数据为：

E7 C7 E7 E7 E7 E7 E7 C3 18 38 18 18 18 18 18 3C

E7	1	1	1	0	0	1	1	1	
	0	0	0	1	1	0	0	0	18
C7	1	1	0	0	0	1	1	1	
	0	0	1	1	1	0	0	0	38
E7	1	1	1	0	0	1	1	1	
	0	0	0	1	1	0	0	0	18
E7	1	1	1	0	0	1	1	1	
	0	0	0	1	1	0	0	0	18
E7	1	1	1	0	0	1	1	1	
	0	0	0	1	1	0	0	0	18
E7	1	1	1	0	0	1	1	1	
	0	0	0	1	1	0	0	0	18
E7	1	1	1	0	0	1	1	1	
	0	0	0	1	1	0	0	0	18
C3	1	1	0	0	0	0	1	1	
	0	0	1	1	1	1	0	0	3C

图 5－4　选择不同颜色“1”的字模数据结构

四、图形库在内存中的位置

把所有字符的字模集合在一起就构成了一个图形库。由于一个字模占用 16 个字节，256 个字节要占用 4K 字节，所以任天堂游戏中的背景图形库容量为 4K 字节。

与背景图形相同，卡通图形也是 256 个，也要占用 4K 字节。

背景、卡通图形库共占用 PPU 所管理内存的 $0000～$1FFF 的 8K 空间。一般使用 8×8K 的 EPROM2764 存放，它共分两个区：第Ⅰ区、第Ⅱ区。

第Ⅰ区占用 $0000～$0FFF4K 空间，存放 256 个 8×8

点阵的卡通图形块(包括字符)。

第Ⅱ区占用$1000～$1FFF4K空间，存放256个8×8点阵的背景图形块(包括字符)。

我们可使用程序No.3－6或No.3－7读出图形库数据以进行研究。

第二节　图形块设计

在图形库中，从$0000开始，每连续的16个字节为一段，定义一个8×8点阵的图形块。

下面通过实例说明图形块的设计过程。

例5－1 定义一个图形块，要求：以"00"号配色为底色；用"01"号配色围成一个方框，在方框内有一个罗马数字"Ⅰ"；要求用"10"号配色写横；用"11"号配色写竖。图形块如图5－5所示。

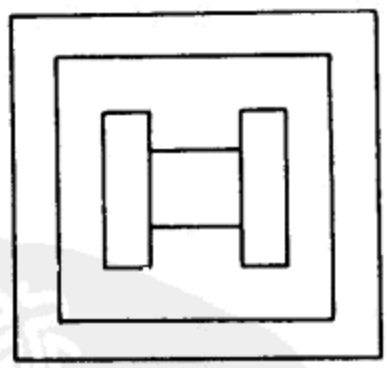


图5－5　罗马数字"Ⅰ"

定义过程如下：

(1)画一个8×8的方格，如图5－6所示。

列：	1	2	3	4	5	6	7	8	
行:00	0	0	0	0	0	0	0	0	
一	1	1	1	1	1	1	1	1	FF
二 00	0	0	0	0	0	0	0	0	
	1	0	0	0	0	0	0	1	81
三 3C	0	0	1	1	1	1	0	0	
	1	0	0	0	0	0	0	1	81
四 18	0	0	0	1	1	0	0	0	
	1	0	0	1	1	0	0	1	99
五 18	0	0	0	1	1	0	0	0	
	1	0	0	1	1	0	0	1	99
六 3C	0	0	1	1	1	1	0	0	
	1	0	0	0	0	0	0	1	81
七 00	0	0	0	0	0	0	0	0	
	1	0	0	0	0	0	0	1	81
八 00	0	0	0	0	0	0	0	0	
	1	1	1	1	1	1	1	1	FF

图 5－6　罗马数字“Ⅰ”字模结构图

(2)把最外圈的小方格中填入“01”号配色代码，使其构成1＃配色的方框。填入数字时，要求一个方格中的数字分两行填写。填入“01”时，上一行填高位“0”、下一行填低位“1”。(下同)

(3)从第三列开始填入四个“10”，构成“Ⅰ”的上横。从第三列第六行开始填入四个“10”，以构成“Ⅰ”的下横。

(4)从第四列第四行开始填入两个“11”，从第四列第五行开始填入两个“11”，以构成“Ⅰ”的一竖。

(5)把空白方格填入“00”。

(6)把每一格中上行的8位二进制数写在方框的左边；把下行的8位二进制数写在方框的右边；

(7)先自上而下抄下左边的8个数据，再抄右边的8个数据。这16个数据就是例5－1的图形数据。它们为：

00 00 3C 18 18 3C 00 00

FF 81 81 99 99 81 81 FF

若以空白“□”代表“00”号配色、以“■”代表“01”号配色、以“※”代表“10”配色、以“●”代表“11”号配色，对图 5－5 进行涂色后，其效果如图 5－7 所示。

图 5－7　罗马数字“Ⅰ”效果图

第三节　卡通字模的结构

卡通图形的设计过程与背景图形大同小异，所不同的是：卡通图形一般都是 16×16 点阵的。一个卡通图形要由四个 8×8 点阵的卡通块组成。因此，在设计卡通图形时，要把一个卡通图形分为四份，每一个卡通块占图形的四分之一。我们先验证一下 F BASIC 中玛丽哥哥的定义数据。

查阅卡通图形库，$0040～$007F 连续的 64 个单元是玛丽走 2 的点阵数据，现抄录如下：

0040－00　00　07　1F　3F　1F　0F　00

0048－03　0F　00　02　04　1E　00　03

0050－00　00　E0　F0　F0　F0　C0　80

0058－C0　E0　E0　50　D0　30　00　E0

0060－01　11　1F　0F　07　01　00　00

0068－07　0F　0F　0E　06　07　0E　01

0070—00　80　80　A0　E0　C0　00　00

0078—E0　E0　60　60　60　C0　C0　C0

把数据填入 4 个 8×8 的方格表中。见图 5—8。

4

00	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	1	1	03
00	0	0	0	0	0	0	0	0	
	0	0	0	0	1	1	1	1	0F
07	0	0	0	0	0	1	1	1	
	0	0	0	0	0	0	0	0	00
1F	0	0	0	1	1	1	1	1	
	0	0	0	0	0	0	0	0	02
3F	0	0	1	1	1	1	1	1	
	0	0	0	0	0	0	0	0	04
1F	0	0	0	1	1	1	1	1	
	0	0	0	1	1	1	1	0	1E
0F	0	0	0	0	1	1	1	1	
	0	0	0	0	0	0	0	0	00
00	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	1	1	03

5

00	0	0	0	0	0	0	0	0	
	1	1	0	0	0	0	0	0	C0
00	0	0	0	0	0	0	0	0	
	1	1	1	0	0	0	0	0	E0
E0	1	1	1	0	0	0	0	0	
	1	1	1	0	0	0	0	0	E0
F0	1	1	1	1	0	0	0	0	
	0	1	0	1	0	0	0	0	50
F0	1	1	1	1	0	0	0	0	
	1	1	0	1	0	0	0	0	D0
F0	1	1	1	1	0	0	0	0	
	0	0	1	1	0	0	0	0	30
C0	1	1	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	00
80	1	0	0	0	0	0	0	0	
	1	1	1	0	0	0	0	0	E0

6

01	0	0	0	0	0	0	0	1	
	0	0	0	0	0	1	1	1	07
11	0	0	0	1	0	0	0	0	
	0	0	0	0	1	1	1	1	0F
1F	0	0	0	1	1	1	1	1	
	0	0	0	0	1	1	1	1	0F
0F	0	0	0	0	1	1	1	1	
	0	0	0	0	1	1	1	0	0E
07	0	0	0	0	0	1	1	1	
	0	0	0	0	0	1	1	0	06
01	0	0	0	0	0	0	0	1	
	0	0	0	0	0	0	1	1	07
00	0	0	0	0	0	0	0	0	
	0	0	0	0	1	1	1	0	0E
00	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	1	01

7

00	0	0	0	0	0	0	0	0	
	1	1	1	0	0	0	0	0	E0
80	1	0	0	0	0	0	0	0	
	1	1	1	0	0	0	0	0	E0
80	1	0	0	0	0	0	0	0	
	0	1	1	0	0	0	0	0	60
A0	1	0	1	0	0	0	0	0	
	0	1	1	0	0	0	0	0	60
E0	1	1	1	0	0	0	0	0	
	0	1	1	0	0	0	0	0	60
C0	1	1	0	0	0	0	0	0	
	1	1	0	0	0	0	0	0	C0
00	0	0	0	0	0	0	0	0	
	1	1	0	0	0	0	0	0	C0
00	0	0	0	0	0	0	0	0	
	1	1	0	0	0	0	0	0	C0

图 5—8　玛丽走 2 的图形数据

若仍按例 5—1 配色所代表色块的约定，对图 5—8 涂色，将得到玛丽走 2 的图形，见图 5—9。

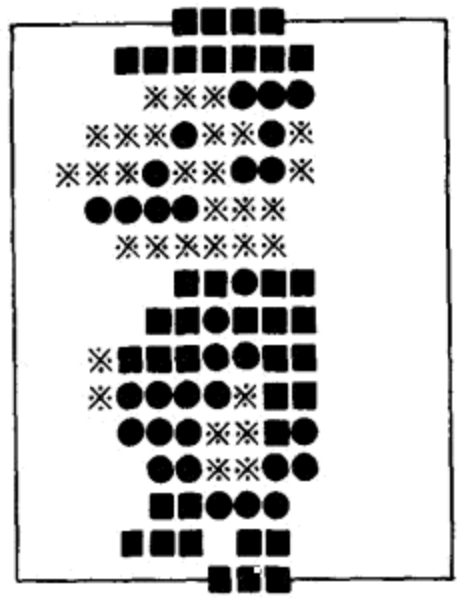


图 5－9　玛丽走 2 图形

显然，读出的卡通图形库字模数据是正确的。

第四节　图形的放大

凡玩过《超级玛丽》游戏的朋友都知道，游戏开始时玛丽只有 16×16 点阵大小，当吃到红蘑菇后，身体立即长大为 32×32 点阵大小。但 F BASIC 中没有这么大的卡通图形，它的图形数据将是怎样的呢？

实际上并不神秘，它不过是把玛丽的 16×16 点阵的图形放大一倍而已。下面就讨论 32×32 点阵玛丽图形的设计方法。

例 5－2 试求 32×32 点阵的玛丽走 2 的图形数据。现在，我们已经掌握了 16×16 点阵的玛丽走 2 的图形数据，那么求 32×32 点阵的图形数据将变的十分容易。凡实践过图形的方格放大的朋友都清楚：当要把图形放大一倍时，只要把

方格的尺寸比原图方格扩大一倍，然后照原图填入每个方格中的内容，就得到了放大了的图象。因此，对例 5－2，只要把原 16×16 点阵图形中的 4 个 8×8 点阵图形块分成 16 个 4×4 点阵块；然后把每一个 4×4 点阵块的数据填入 8×8 点阵中，即 4×4 点阵中的一个方格中的数据要填入 8×8 点阵中 4 个相邻的方格中；最后把这 16 个 8×8 点阵的图形块组合在一起，就完成了 32×32 点阵玛丽走 2 的图形设计。

以上介绍了图形放大的思路，实际放大操作中一般并不直接使用方格放大(哪样填写 16 个 8×8 点阵方格图的工作量太大，一般采用计算法模拟方格放大。

下面我们仅以原图形库中玛丽走 2 的第一个 8×8 点阵块放大为例，介绍计算法图形放大的操作步骤。

1. 读出玛丽走 2 的第一个图形块数据(即图形库第 I 区，序号为 4 的图形数据)。这 16 个数据为：

40— 00 00 07 1F 3F 1F 0F 00

48— 03 0F 00 02 04 1E 00 03

2. 把以上 16 个数据分成 4 组，每组 4 个，分别排列如下：

(1)00 00 07 1F

(2)03 0F 00 02

(3)3F 1F 0F 00

(4)04 1E 00 03

3. 把以上 4 组数据中的每一个数据按高、低位拆开，高位为一组、低位为一组，共组成八组。排列如下：

$$A\begin{bmatrix}(1)0\ 0\ 0\ 1\\(2)0\ 0\ 0\ 0\end{bmatrix}\qquad B\begin{bmatrix}(1)'\ 0\ 0\ 7\ F\\(2)'\ 3\ F\ 0\ 2\end{bmatrix}$$

C [(3)3 1 0 0
(4)0 1 0 0]　　D [(3)' F F F 0
(4)' 4 E 0 3]

4. 把每一组数字按表 5—1 的规则替换成相应的数据。

表 5—1　图形放大数据换算表

原始数据	替换数据	原始数据	替换数据
0	00	8	C0
1	03	9	C3
2	0C	A	CC
3	0F	B	CF
4	30	C	F0
5	33	D	F3
6	3C	E	FC
7	3F	F	FF

以上各组替换完毕后的数据如下：

A [(1)00 00 00 03
　 00 00 00 00]

B [(1)' 00　00 3F FF
(2)' 0F FF 00 0C]

C [(3)0F 03 00 00
(4)00 03 00 00]

D [(3)' FF FF FF 00
(4)' 30 FC 00 0F]

5. 把以上各组中的数据每一个重写一次，使每行由 4 个数据变为 8 个数据。则有：

A [(1)00 00 00 00 00 00 03 03
(2)00 00 00 00 00 00 00 00]

B [(1)' 00 00 00 00 3F 3F FF FF
(2)' 0F 0F FF FF 00 00 0C 0C]

C (3)0F 0F 03 03 00 00 00 00
(4)00 00 03 03 00 00 00 00

D (3)' FF FF FF FF FF FF 00 00
(4)' 30 30 FC FC 00 00 0F 0F

以上 A、B、C、D 四组数据，即为 4 号卡通块放大一倍后的图形数据。现在，我们把它写入方格验证一下。

验证情况见图 5－10、图 5－11。

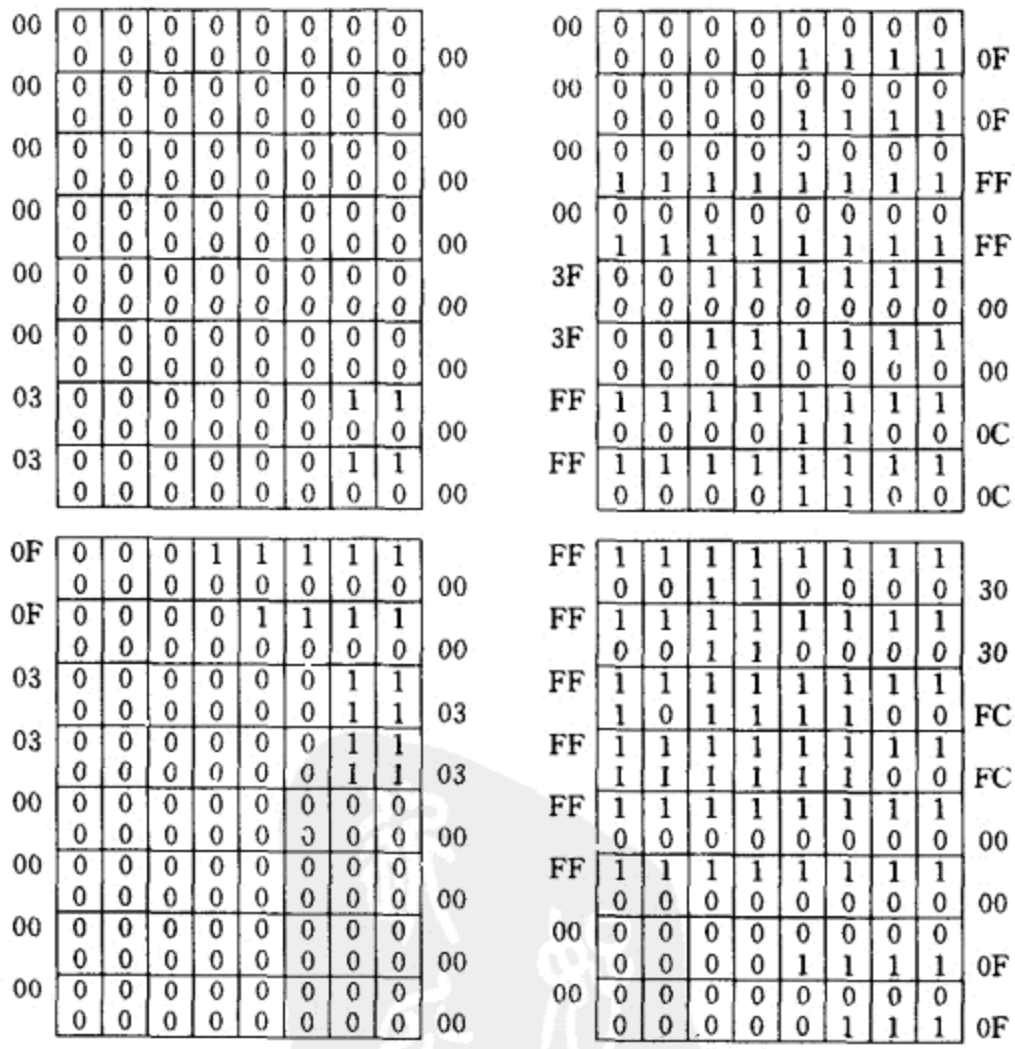


图 5－10　4＃卡通(玛丽走 2)放大一倍后的图形数据

可见，放大的图形数据都是正确的。

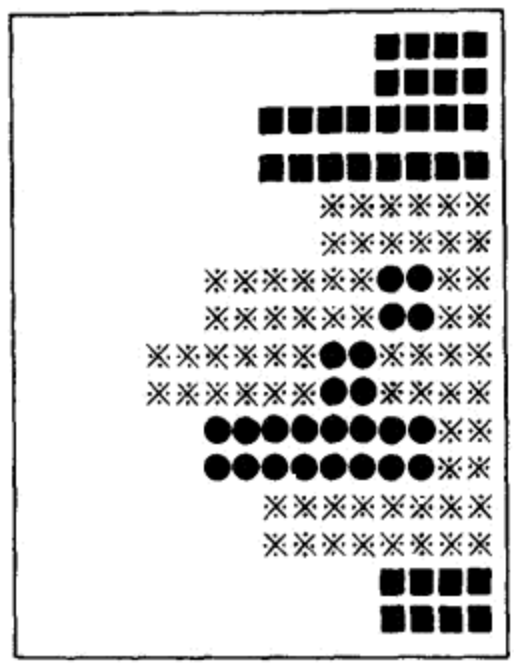


图 5－11　4# 卡通(玛丽走 2)放大一倍后的图形

仿照以上五个步骤，可以写出后三个卡通块 5#、6#、7#的放大数据。有兴趣的读者可以自己做一遍。下面仅给出最后的结果(以 A、B、C……为序)：

E
```
00 00 00 00 FC FC FF FF
F0 F0 FC FC FC FC 33 33
```

F
```
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
```

G
```
FF FF FF FF F0 F0 C0 C0
F3 F3 0F 0F 00 00 FC FC
```

H
```
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
```

I
```
00 00 03 03 03 03 00 00
00 00 00 00 00 00 00 00
```

J
```
03 03 03 03 FF FF FF FF
3F 3F FF FF FF FF FC FC
```

K
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00

L
3F 3F 03 03 00 00 00 00
3C 3C 3F 3F FC FC 03 03

M
00 00 C0 C0 C0 C0 CC CC
FC FC FC FC 3C 3C 3C 3C

N
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00

O
FC FC F0 F0 00 00 00 00
3C 3C F0 F0 F0 F0 F0 F0

P
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00

读者可把以上 12 组数据填入 12 个 8×8 方格中，以对比放大后的效果。

第五节　自制图形的使用

通过以上讨论，我们基本弄通了图形块设计的方法步骤。同时也解释了我们的电脑游戏机中为什么没有《魂斗罗》、《双截龙》等游戏中的背景画面和卡通的缘故。明白了这一道理后，我们可以随意设计自己喜欢、需要的图形字模数据了。

但是，图形字模数据设计完毕后，应该怎样使用自己制造的图形呢？

目前生产的电脑学习机，系统 ROM 中固化了多个软件，软件不同，各个软件使用的字模数据也不相同。为了降低成本、简化硬件结构，多数机型使用了 8K 静态 RAM(6264)存储器作为字模存储器，而把各系统的字模数据存放在系统 ROM 中。当进入某一系统时，首先把该系统的字模数据送入

6264，然后再转入系统复位程序入口。这样为字模数据的更换提供了方便(许多任天堂游戏也采用这种工作方式，如《魂斗罗》一代、《绿色兵团》等)，也为我们使用自己设计的图形提供了条件。

使用自己设计图形的一般方法是：设计好字模数据后，首先为该图形分配图形代码(代码的选择可根据实际需要，优先选择那些不常使用的代码，如背景图形库中的日文假名、卡通图形库中的羽毛笔、"COMPUTER、OPERATOR"等)；然后用写 PPU 的方法，把数据写入指定代码的字模地址单元。这样操作完毕，只要不按复位键，自己设计的图形可一直使用。有条件的读者，可把图形数据写入程序中，把程序保存起来，需要时调出程序运行一次，就具备自造图形了。

本节仅讨论了图形字模设计的一般方法，在实际的图形设计中，一般使用工具软件，让电脑把设计好的图形自动转换成图形数据或者截取、借鉴其它游戏软件的图形数据，以提高图形设计效率。这些问题已超出本书的讨论范畴，恕不赘述。

第六节　电脑学习机的显示方式

通常，把直接对 PPU 的显示映射区操作、每对一个单元操作就可显示一个 8×8 点阵的字符或图形的显示方式，称为"字符显示方式"。在这种显示方式下，字模区数据固定不变，屏幕显示映射区的内容改变。显示操作中，用户不必理会字模数据的组成，只要向显示映射区置入代码就可以实现字符或图形显示。因此，这种显示方式处理简便、显示速度快捷，任天堂游戏就采用了这种显示方式。

但是，字符显示方式有一个先天缺陷，就是所显示的字符种类有限制(限定在256个)，当超过限制时，就要重送字模数据以改变字模的种类。重送字模时，必须处理好当前屏幕上显示的内容所对应的字模种类，不能因字模数据的改变而影响了已显示的内容，即要考虑字模的兼容性。这样做对字模种类超过限制不太多时还好处理，当超出很多，如显示一、二级字库中的六千余个汉字时，麻烦就大了。因此，电脑学习机中对汉字的显示又采用了另外的处理方式——图形显示方式和准图形显示方式。

一、图形显示方式

图形显示方式的工作模式与字符显示方式相反。在图形显示方式下，PPU 显示映射区的数据固定不变，字模区的数据改变。这样，当欲在屏幕上显示某个汉字时，只须取出汉字的字模送入显示位置对应的代码字模区即可。

图形显示方式是通过改造硬件结构实现的。与字符显示方式相比，主要有以下几点区别。

1. 字模数据结构

在图形方式下，每个8×8点阵图形的字模数据又恢复为8个，即屏幕上每个点的显示状态由一位二进制数表示。这样使字符显示方式下的一个字符字模位置可显示两个字符。因此，原4K空间的字模区可以保存512个字符的字模。为了仍然使用1个字节表示字模代码，把原字符方式下的一个代码分配给现在对应地址的两个字符。如字符方式下的背景图形代码41H(字符A)的对应PPU字模地址为$1410～$141F，现在被分配给两个字符，这两个字符的代码都为41H，但有前41H、后41H之分。因此，背景字模区的图形代

码依次为

00 00 01 01 02 02 03 03 04 04 …… FF FF

2. 字模数据库的结构

在图形方式下，原卡通字模数据库也被用来存背景的字模，字模数据结构与背景区相同。图形代码也依次为

00 00 01 01 02 02 03 03 04 04 …… FF FF

这样，8K 字模区可以存放 1024 个字符的字模。

3. PPU 显示映射区的结构

在字符方式下，PPU 显示映射区的数据是根据显示内容而随时改变的，清屏时，各单元存入空格代码(如在 F BASIC 状态下，存入 20H)。而图形显示方式下，PPU 显示映射区的数据是固定的，数据结构如图 5－12 所示。

图 5－12 中的数据结构有两个令人费解的地方：

(1)从 PPU 地址 $2000 开始，每两个显示单元中的图形代码相同，是否相同的代码就显示相同的内容呢？

答案是：代码相同，显示的内容可以不同(也允许相同)。其实，两个相同的代码分别对应着字模区的前、后 8 个字节的字模数据。如地址 $2262、$2263 中的代码都是 41H，它们就对应着字模区的前 41H 和后 41H。

(2)前 15 行与后 15 行的数据对应相同，是否上半屏显示的内容与下半屏相同呢？

答案是：上、下半屏显示的内容各不相同(也允许相同)。实际上，上半屏的图形代码对应着 PPU 字模区 $0000～$0EFF 单元的字模数据，下半屏则对应着字模区 $1000～$1EFF 单元的字模数据。

列地址	00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F
P2000	00 00 01 01 02 02 03 03 04 04 05 05 06 06 07 07 08 08 09 09 0A 0A 0B 0B 0C 0C 0D 0D 0E 0E 0F 0F
P2020	10 10 11 11 12 12 13 13 14 14 15 15 16 16 17 17 18 18 19 19 1A 1A 1B 1B 1C 1C 1D 1D 1E 1E 1F 1F
U2040	20 20 21 21 22 22 23 23 24 24 25 25 26 26 27 27 28 28 29 29 2A 2A 2B 2B 2C 2C 2D 2D 2E 2E 2F 2F
行 2060	30 30 31 31 32 32 33 33 34 34 35 35 36 36 37 37 38 38 39 39 3A 3A 3B 3B 3C 3C 3D 3D 3E 3E 3F 3F
地 2080	40 40 41 41 42 42 43 43 44 44 45 45 46 46 47 47 48 48 49 49 4A 4A 4B 4B 4C 4C 4D 4D 4E 4E 4F 4F
址 20A0	50 50 51 51 52 52 53 53 54 54 55 55 56 56 57 57 58 58 59 59 5A 5A 5B 5B 5C 5C 5D 5D 5E 5E 5F 5F
20C0	60 60 61 61 62 62 63 63 64 64 65 65 66 66 67 67 68 68 69 69 6A 6A 6B 6B 6C 6C 6D 6D 6E 6E 6F 6F
20E0	70 70 71 71 72 72 73 73 74 74 75 75 76 76 77 77 78 78 79 79 7A 7A 7B 7B 7C 7C 7D 7D 7E 7E 7F 7F
2100	80 80 81 81 82 82 83 83 84 84 85 85 86 86 87 87 88 88 89 89 8A 8A 8B 8B 8C 8C 8D 8D 8E 8E 8F 8F
2120	90 90 91 91 92 92 93 93 94 94 95 95 96 96 97 97 98 98 99 99 9A 9A 9B 9B 9C 9C 9D 9D 9E 9E 9F 9F
2140	A0 A0 A1 A1 A2 A2 A3 A3 A4 A4 A5 A5 A6 A6 A7 A7 A8 A8 A9 A9 AA AA AB AB AC AC AD AD AE AE AF AF
2160	B0 B0 B1 B1 B2 B2 B3 B3 B4 B4 B5 B5 B6 B6 B7 B7 B8 B8 B9 B9 BA BA BB BB BC BC BD BD BE BE BF BF
2180	C0 C0 C1 C1 C2 C2 C3 C3 C4 C4 C5 C5 C6 C7 C7 C8 C8 C9 C9 CA CA CB CB CC CC CD CD CE CE CF CF
21A0	D0 D0 D1 D1 D2 D2 D3 D3 D4 D4 D5 D5 D6 D6 D7 D7 D8 D8 D9 D9 DA DA DB DB DC DC DD DD DE DE DF DF
21C0	E0 E0 E1 E1 E2 E2 E3 E3 E4 E4 E5 E5 E6 E6 E7 E7 E8 E8 E9 E9 EA EA EB EB EC EC ED ED EE EE EF EF
21E0	00 00 01 01 02 02 03 03 04 04 05 05 06 06 07 07 08 08 09 09 0A 0A 0B 0B 0C 0C 0D 0D 0E 0E 0F 0F
2200	10 10 11 11 12 12 13 13 14 14 15 15 16 16 17 17 18 18 19 19 1A 1A 1B 1B 1C 1C 1D 1D 1E 1E 1F 1F
2220	20 20 21 21 22 22 23 23 24 24 25 25 26 26 27 27 28 28 29 29 2A 2A 2B 2B 2C 2C 2D 2D 2E 2E 2F 2F
2240	30 30 31 31 32 32 33 33 34 34 35 35 36 36 37 37 38 38 39 39 3A 3A 3B 3B 3C 3C 3D 3D 3E 3E 3F 3F
2260	40 40 41 41 42 42 43 43 44 44 45 45 46 46 47 47 48 48 49 49 4A 4A 4B 4B 4C 4C 4D 4D 4E 4E 4F 4F
2280	50 50 51 51 52 52 53 53 54 54 55 55 56 56 57 57 58 58 59 59 5A 5A 5B 5B 5C 5C 5D 5D 5E 5E 5F 5F
22A0	60 60 61 61 62 62 63 63 64 64 65 65 66 66 67 67 68 68 69 69 6A 6A 6B 6B 6C 6C 6D 6D 6E 6E 6F 6F
22C0	70 70 71 71 72 72 73 73 74 74 75 75 76 76 77 77 78 78 79 79 7A 7A 7B 7B 7C 7C 7D 7D 7E 7E 7F 7F
22E0	80 80 81 81 82 82 83 83 84 84 85 85 86 86 87 87 88 88 89 89 8A 8A 8B 8B 8C 8C 8D 8D 8E 8E 8F 8F
2300	90 90 91 91 92 92 93 93 94 94 95 95 96 96 97 97 98 98 99 99 9A 9A 9B 9B 9C 9C 9D 9D 9E 9E 9F 9F
2320	A0 A0 A1 A1 A2 A2 A3 A3 A4 A4 A5 A5 A6 A6 A7 A7 A8 A8 A9 A9 AA AA AB AB AC AC AD AD AE AE AF AF
2340	B0 B0 B1 B1 B2 B2 B3 B3 B4 B4 B5 B5 B6 B6 B7 B7 B8 B8 B9 B9 BA BA BB BB BC BC BD BD BE BE BF BF
2360	C0 C0 C1 C1 C2 C2 C3 C3 C4 C4 C5 C5 C6 C6 C7 C7 C8 C8 C9 C9 CA CA CB CB CC CC CD CD CE CE CF CF
2380	D0 D0 D1 D1 D2 D2 D3 D3 D4 D4 D5 D5 D6 D6 D7 D7 D8 D8 D9 D9 DA DA DB DB DC DC DD DD DE DE DF DF
23A0	E0 E0 E1 E1 E2 E2 E3 E3 E4 E4 E5 E5 E6 E6 E7 E7 E8 E8 E9 E9 EA EA EB EB EC EC ED ED EE EE EF EF

图 5－12　图形显示方式下的 PPU 显示映射区数据结构示意图

以上两个相同的前后代码的区分、两个相同代码的上下屏的区分都是由硬件电路控制完成的。

4. 空余图形代码的使用

从图 5－12 可见，图形显示方式下还有两组 F0～FFH 共 32 个代码没有使用，这 32 个代码可用于定义卡通。

5. 图形方式下的字符显示操作

图形方式下字符的显示操作如下：

(1)清屏

因在图形方式下，屏幕上每个点的状态只有亮、灭两种，故清屏操作就是把字模区 $0000～$1FFF 的所有单元置入 0 即可。

(2)显示字符

显示字符的操作,就是把该字符的字模数据,送入显示位置的图形代码对应的字模地址中去。如欲在屏幕的第二行第三列显示“1”,则把“1”的8个字模数据置入字模区$0110连续的8个单元中。如欲在屏幕的第四行第四列显示“A”,则把“A”的8个字模数据置入字模区$0318开始的8个单元中(因第四列为后31H,所以应送入31H对应的后8个字节单元)。再如欲在屏幕的第20行第1列显示字符“@”,由图5—12可知该单元对应的图形代码为40H,因是下半屏,故“@”的字模数据应置入$1400开始的8个单元中。

(3)显示汉字

因一个汉字由4个8×8点阵的字符组成,即一个汉字的字模数据为32个字节,所以应取出该汉字的32个字模数据送入各图形块对应的字模地址单元。如欲在屏幕的第四行第六列显示一个汉字,该显示位置对应的图形代码为:

32 33

42 43

故应取出汉字左上角的8个字模数据送入$0328开始的8个单元(因是后32H)、取出右上角的8个字模数据送入$0330开始的8个单元(因是前33H)、取出左下角的8个字模数据送入$0428开始的8个单元、取出右下角的8个字模数据送入$0430开始的8个单元。

二、图形显示方式的特点

1. 可全屏幕显示汉字

由图5—12可见,在图形方式下,可以全屏幕显示汉字,显示汉字的极限值为240个(16列×15行)。因此,显示方式

对软件设计没有制约。

2. 可按点操作

在图形显示方式下，用户可以对屏幕按点操作，可以控制每个点的亮与灭，因而可以显示较细腻的图形，LOGO 语言就是依赖于这种显示方式工作的。

3. 色彩单一

全屏幕显示汉字的功能是牺牲色彩换来的。因此，图形显示方式下，汉字的色彩较单一，虽然能显示 4 种颜色的字符，但不能在一个字符中显示多种颜色。

4. 硬件结构复杂

由于屏幕显示和字模区结构的特殊要求，图形显示方式的硬件结构复杂。

5. 系统的显示功能强

任天堂游戏机的基本显示方式就是字符方式，若再加入图形显示方式后，将使系统的显示功能增强一倍。使用更加灵活，可以运行任何显示方式的软件（因显示方式的切换，只需对软开关置数就可方便地实现。使用中感觉不到切换过程）。

6. 具有图形显示方式的机型

由于挂接图形显示方式要增加硬件电路，相应地增加制造成本，故只有高档学习机才采用。较有代表性的有“裕兴”和“金字塔”。

三、准图形显示方式

以“小霸王”为代表的低档电脑学习机，如“科达”、“科王”、“美猴王”、“王中王”等机型，为了维持其低成本、低价位的特点，在汉字的显示方式上采用了在软件方面下功夫的“准图形显示方式”。

1. 准图形显示方式的字模结构

准图形显示方式也把屏幕上的点改为两种状态(亮、灭)，也是使用 8 个字模数据表示一个字符。

2. 准图形显示方式的字模区结构

准图形显示方式也使用了 8K 字模区，它的每个 8×8 点阵的字符仍使用 16 个字节，其中前 8 个字节存字模数据，后 8 个字节空闲不用(置入 0)。因此，8K 字模区可以容纳 512 个字符，即 128 个汉字的字模。

3. 准图形显示方式的屏幕显示映射区结构

由于准图形显示方式的字模区最多能容纳 128 个汉字，

列地址	000102030405060708090A0B0C0D0E0F101112131415161718191A1B1C1D1E1F
P 2000	..01020304050 6.....10111213.....14151617..
P 2020	..18191A1B1C1D..07080 90A0B0C0D0E0F....27282 92A.....2B2C2D2E..
U 2040	..2626262626262626262626262626262626262626262626262626262626..
行 2060	.....303132333435363738393A3B3C3D3E3F404142434445464748 49.....
地 2080	.....4A4B4C4D4E4F505152535455565758595A5B5C5D5E5F606162 63.....
址 20A0	.....
20C0	.....646566676869 6A6B6C6D6E6F707172737475767778797A7B7C7D.....
20E0	.....7E7F808182838485868788898A8B8C8D8E8F909192939495969 7.....
2100	.....
2120	.....98999A9B9C9D9E9FA0A1A2A3A4A5A6A7A8A9AAABACADAEAFB0B1.....
2140	.....B2B3B4B5B6B7B8B9BABBBCBDBEBFC0C1C2C3C4C5C6C7C8C9CACB.....
2160	.....
2180	.....CCCDCECFD0D1D2D3D4D5D6D7D8D9DADBDCDDDEDFE0E1E2E3E4E5.....
21A0	.....E6E7E8E9EAEBECEDEEEFF0F1F2F3F4F5F6F7F8F9FAFBFCFDFEFF.....
21C0	.....
21E0	.....303132333435363738393A3B3C3D3E3F404142434445464748 49.....
2200	.....4A4B4C4D4E4F505152535455565758595A5B5C5D5E5F606162 63.....
2220	.....
2240	.....646566676869 6A6B6C6D6E6F707172737475767778797A7B7C7D.....
2260	.....7E7F808182838485868788898A8B8C8D8E8F909192939495969 7.....
2280	.....
22A0	.....616262636364646565666667676868696 96A6A6B6B6C6C6D6D6E.....
22C0	.....717272737373747475757676777778787979 7A7A7B7B7C7C7D7D7E.....
22E0	.....818282838383848485858686878788888989 8A8A8B8B8C8C8D8D8E.....
2300	.....CCCDCECFD0D1D2D3D4D5D6D7D8D9DADBDCDDDEDFE0E1E2E3E4E5.....
2320	.....E6E7E8E9EAEBECEDEEEFF0F1F2F3F4F5F6F7F8F9FAFBFCFDFEFF.....
2340	.....
2360	..2A2A2A2A2A2A2A2A2A2A2A2A2A2A2A2A2A2A2A2A2A2A2A2A2A2A2A2A..
2380	..01020304.....09181 9.....0E1A1B.....131C1D.....
23A0	..2B2C2D2E05060708....1E1F0A0B0C0D..20210F10111 2..22231415161 7..

注——为便于观察，图中把数据"00"以".."代替。

图 5－13　准图形显示方式的屏幕显示映射区数据结构示意图

这就决定了准图形显示方式的屏幕上最多只能显示128个汉字。若再除去某些屏幕提示(如中文编辑中的文件名、汉字输入法提示、重码汉字提示等)，一般只能显示汉字100个左右。准图形显示方式屏幕显示映射区的数据结构如图5－13所示。

4. 准图形显示方式的工作原理

由图5－13可见，准图形显示方式只能显示256个西文字符。在编辑工作区(即PPU地址$2063～$233C的区域)只能显示104个汉字。每空一个8点阵行显示一行汉字并不是为了便于阅读，而是不得不这样安排。

准图形显示方式的显示屏幕分上下两个半屏，上半屏使用字模图形库$1000～$1FFF中的数据，下半屏使用$0000～$0FFF中的数据。当需要在某一位置显示汉字时，则向屏幕上显示单元对应的字模区送入字模数据。

由于上、下半屏使用的字模数据不同，故必须在画面形成过程中采取“画面分裂”技术(有关画面分裂的概念将在第八章讨论)。一般画面的形成在中断中进行，“小霸王”的中断处理程序如下：

```
        中断处理程序
9F08 48   PHA ;┐
9F09 8A   TXA  │
9F0A 48   PHA  │ 保护现场
9F0B 98   TYA  │
9F0C 48   PHA  │
9F0D 08   PHP ;┘
```

```
9F0E A900      LDA #$00;┐
9F10 8D0320    STA $2003 │
9F13 A902      LDA #$02  │ 置卡通定义区
9F15 8D1440    STA $4014;┘
9F18 ADFB7F    LDA $7FFB;┐ $7FFB 为中断处理控制开
9F1B F003      BEQ $9F20 │ 关，$7FFB≠0 时直接返回
9F1D 4CE99F    JMP $9FE9;┘
9F20 A5CB      LDA $CB  ;┐ 以 $CB 为开关
9F22 F006      BEQ $9F2A │ 以 $D5 变址置
9F24 205DBA    JSR $BA5D │ PPU 地址，向 PPU 送 39H
9F27 4C7E9F    JMP $9F7E;┘ 个数据
9F2A A5C7      LDA $C7;  ┐
9F2C F00A      BEQ $9F38 │
9F2E 20A905    JSR $05A9 │ 以 $C7 为开关
9F31 A900      LDA #$00  │ 送显一个汉字
9F33 85C7      STA $C7   │
9F35 4C7E9F    JMP $9F7E;┘
9F38 A5CF      LDA $CF;  ┐
9F3A F01F      BEQ $9F5B │
9F3C A901      LDA #$01  │
9F3E 85EA      STA $EA   │
9F40 200005    JSR $0500 │ 以 $CF 为开关
9F43 A527      LDA $27   │ 一次送显两个西文字符
9F45 85E2      STA $E2   │
9F47 2098A8    JSR $A898 │
9F4A 206DB6    JSR $B66D │
9F4D A901      LDA #$01  ┘
```

```
9F4F 85EA     STA $EA
9F51 200005   JSR $0500
9F54 A900     LDA #$00
9F56 85CF     STA $CF
9F58 4C7E9F   JMP $9F7E;
9F5B 20EFB4   JSR $B4EF    ;由$C5 为开关控制光标闪烁
9F5E A5C1     LDA $C1;
9F60 F00A     BEQ $9F6C
9F62 A901     LDA #$01     以$C1 为开关
9F64 85EA     STA $EA      送显一个西文字符
9F66 200005   JSR $0500
9F69 4C7E9F   JMP $9F7E;
9F6C A5C2     LDA $C2;
9F6E F007     BEQ $9F77
9F70 A901     LDA #$01     以$C2 为开关
9F72 85EA     STA $EA      向 PPU 送 8 个数据
9F74 201F05   JSR $051F;
9F77 A5C3     LDA $C3;
9F79 F003     BEQ $9F7E    以$C3 为开关
9F7B 208A05   JSR $058A;   向 PPU 送半个汉字的字模
9F7E A900     LDA #$00
9F80 8D0520   STA $2005
9F83 8D0520   STA $2005
9F86 A990     LDA #$90
9F88 8D0020   STA $2000    ;背景使用 1 区字模
9F8B A980     LDA #$80
9F8D 20A1A2   JSR $A2A1    ;延时,形成上半屏画面
```

```
9F90 A988      LDA #$88
9F92 8D0020    STA $2000 ;背景使用0区字模
9F95 A985      LDA #$85
9F97 20A1A2    JSR $A2A1 ;延时,形成下半屏画面
9F9A A5CE      LDA $CE;  ┐以$CE为开关
9F9C F003      BEQ $9FA1 │清除状态行的汉字
9F9E 20B4AD    JSR $ADB4;┘重码提示
9FA1 A5C4      LDA $C4 ; ┐以$C4为开关
9FA3 F003      BEQ $9FA8 │控制部分清屏
9FA5 208AAD    JSR $AD8A;┘
9FA8 A5C9      LDA $C9;  ┐以$C9为开关
9FAA F003      BEQ $9FAF │清除1AH个字符的字模
9FAC 201AA5    JSR $A51A;┘
9FAF A5C8      LDA $C8;  ┐以$C8为开关
9FB1 F003      BEQ $9FB6 │接上面的程序继续清除
9FB3 2044A5    JSR $A544;┘1AH个字符的字模
9FB6 A5C6      LDA $C6;  ┐以$C6为开关
9FB8 F003      BEQ $9FBD │对提示行操作
9FBA 20EEA7    JSR $A7EE;┘
9FBD A5CA      LDA $CA;  ┐
9FBF F007      BEQ $9FC8 │以$CA为开关
9FC1 207405    JSR $0574 │调整数据区指针等
9FC4 A900      LDA #$00  │
9FC6 85CA      STA $CA;  ┘
9FC8 E6DD      INC $DD
```

```
9FCA A5CD    LDA $CD;
9FCC F018    BEQ $9FE6
9FCE A901    LDA #$01
9FD0 8D0440  STA $4004
9FD3 A908    LDA #$08
9FD5 8D0540  STA $4005
9FD8 A9EF    LDA #$EF
9FDA 8D0640  STA $4006
9FDD A90A    LDA #$0A
9FDF 8D0740  STA $4007
9FE2 A900    LDA #$00
9FE4 85CD    STA $CD
9FE6 8D0440  STA $4004;
```

以$CD为开关控制第二主声部发声一次

```
9FE9 28      PLP;
9FEA 68      PLA
9FEB A8      TAY
9FEC 68      PLA
9FED AA      TAX
9FEE 68      PLA;
```

恢复现场

```
9FEF 40      RTI
```

延时子程序

```
A2A1 A215    LDX #$15
A2A3 CA      DEX
A2A4 D0FD    BNE $A2A3
A2A6 38      SEC
A2A7 E901    SBC #$01
A2A9 D0F6    BNE $A2A1
A2AB 60      RTS
```

从中断处理程序可以看出，尽管处理程序很长，但由于

一系列的开关控制，每次中断处理中所执行的程序并不多。

5. 准图形显示方式的特点

(1)不增加成本

由于准图形显示方式纯粹由软件方式实现，故不须改动硬件电路，不增加机器成本。

(2)软件编写受限制

准图形显示方式受字符代码数量的限制，不能满屏显示汉字。因此，软件编写的限制性较大，屏幕上哪些位置可以显示汉字，哪些位置不能显示汉字，必须周密规划。

(3)屏幕滚动不连续

由于在中断中对上下两个半屏切换，当屏幕滚动时，容易产生滞涩的现象。

第六章　背景画面的绘制

以上我们曾讨论了背景显示系统的操作原理，并介绍了背景画面显示、位移的基本编程方法。本章将在以上讨论的基础上，较全面地介绍任天堂游戏中常用的背景画面绘图方法和编程技巧，并给出具体的、独立的应用子程序，供编程中调用。

第一节　局部画面的显示

局部画面显示是任天堂游戏中最常用的一种显示技术，如游戏中角色的生命值、人数、关次的提示、游戏得分的累积、游戏进程的标志、角色对话等内容的显示，都属于局部显示范畴。一般这种形式的显示操作过程，都是在画面的正常显示下进行的，所以不能采取关显示的方法，只能在中断期间(场回扫期间)把显示内容送入 PPU 的显示单元。本节将通过实例具体介绍几种常用的局部绘图方法。

一、直接送显法

例 6－1　在屏幕的左上角(从 PPU 地址 $2005 开始)显示数字“1000”。

实现程序见 No. 6－1

No. 6－1 局部画面送显子程序 (一)

```
7EF1   A9 FA        LDA # $FA
```

```
7EF3    85 EE       STA  $EE
7EF5    A9 7E       LDA  #$7E
7EF7    85 EF       STA  $EF
7EF9    60          RTS
7EFA    20 00 7F    JSR  $7F00
7EFD    4C 0F 88    JMP  $880F
7F00    A9 20       LDA  #$20
7F02    8D 06 20    STA  $2006
7F05    A9 05       LDA  #$05
7F07    8D 06 20    STA  $2006
7F0A    A9 31       LDA  #$31
7F0C    8D 07 20    STA  $2007
7F0F    A9 30       LDA  #$30
7F11    8D 07 20    STA  $2007
7F14    8D 07 20    STA  $2007
7F17    8D 07 20    STA  $2007
7F1A    60          RTS
```

只要把 No. 6－1 放入中断处理程序中，即可送显例 6－1 的数据。读者可参照编写 F BASIC 语言的写入内存程序，写入完毕只要执行“CALL $7EF1”，就可完成送显。（注——在以后的讨论中，为压缩篇幅，除特殊情况外，一般不再列举分 F BASIC 的写入程序。）

但是，在实际应用中，送显的数据往往是变化的，由于我们不可能在程序运行时反复改变中断程序，所以 No. 6－1 仅仅适用于送显数据不变的场合。当送显诸如积分类变化的数字时，应采用例 6－2 的程序 No. 6－2。

例 6－2 在屏幕的右上角（从 PPU 地址 $2015 开始）显

示游戏者的四位即时积分。

我们选取 $A0～$A3 四个连续的零页单元依次存放个、十、百、千四位数据，实现程序见 No. 6－2。

No. 6－2 局部画面送显子程序（二）

```
7EF1   A9 FA       LDA #$FA
7EF3   85 EE       STA $EE
7EF5   A9 7E       LDA #$7E
7EF7   85 EF       STA $EF
7EF9   60          RTS
7EFA   20 00 7F    JSR $7F00
7EFD   4C 0F 88    JMP $880F
7F00   8A          TXA
7F01   48          PHA
7F02   A9 20       LDA #$20
7F04   A2 15       LDX #$15
7F06   20 F8 81    JSR $81F8
7F09   A2 03       LDX #$03
7F0B   B5 A0       LDA $A0,X
7F0D   8D 07 20    STA $2007
7F10   CA          DEX
7F11   10 F8       BPL $7F0B
7F13   68          PLA
7F14   AA          TAX
7F15   60          RTS
```

运行程序 No. 6－2 后，只要向 $A0～$A3 单元置入数据，就可立即显示得分。当欲在画面上连续显示多个内容时，只要选取相应的连续单元数，修改 $7F09 行中变址寄存器

X 的初值即可。但是，当显示的内容较多时，不可能有这么多连续的零页单元供我们使用，所以还是采用查表法为宜。

二、查表送显法

例 6－3 使用查表法局部送显图 6－1 所示的内容。

2000	TIME	SCORE	STAGE	LEVEL
2020	08:35	01234	3	8

图 6－1　局部送显内容

查表法局部送显程序见 No. 6－3。

No. 6－3 局部送显子程序(三)

```
7EF1   A9 FA       LDA # $FA
7EF3   85 EE       STA $EE
7EF5   A9 7E       LDA # $7E
7EF7   85 EF       STA $EF
7EF9   60          RTS
7EFA   20 00 7F    JSR $7F00
7EFD   4C 0F 88    JMP $880F
7F00   8A          TXA
7F01   48          PHA
7F02   A9 20       LDA # $20
7F04   A2 01       LDX # $01
7F06   20 F8 81    JSR $81F8
7F09   A2 00       LDX # $00
7F0B   BD 21 7F    LDA $7F21,X
7F0E   F0 06       BEQ $7F16
```

```
7F10    8D 07 20        STA  $2007
7F13    E8              INX
7F14    10 F5           BPL  $7F0B
7F16    68              PLA
7F17    AA              TAX
7F18    60              RTS
7F20    54 49 4D 45 20 20 20 20 53 43 4F 52 45 20 20
                        T I M E     S C O R E
7F30    20 20 53 54 41 47 45 20 20 20 4C 45 56 45 4C 20
                        S T A G E   L E V E L
7F40    20 30 38 3A 33 35 20 20 20 30 31 32 33 34 20 20
                        0 8 : 3 5   0 1 2 3 4
7F50    20 20 20 20 33 20 20 20 20 20 20 20 38 00
```

程序 No. 6－3 是靠读出为零的数据而结束送数的，它不仅能够进行局部绘图，而且通过简单的扩充数据区就能够绘出整幅图形。因此，程序 No. 6－3 是一个实用性极强的局部绘图程序。

三、开关变址送显法

以上介绍的绘图子程序都必须安装在中断处理程序中运行，也就是说：每执行一次中断，就要把它们的内容送显一次。实际上大多数送显内容不可能变化的那么快，如时间的显示，一秒钟才改变一次，那么在一秒中的五十次中断处理中有四十九次的送显是重复的，这无疑增大了中断处理时间的开销。若中断处理程序较短小则影响不大，当中断程序量较大时，就有可能造成一个场回扫时间不够用的情况，从而造成显示紊乱。因此，一般在中断处理的各子程序开始设置一个执行

软开关，只有开关打开时才执行该子程序，当开关关闭时则跳过该子程序执行其它中断程序。这样处理可减少中断执行时间的开销，加快中断执行速度，提高中断工作效率。

另外在实际局部送显中，常有大部分内容不变，少部分内容改变的情况，如图 6－1 中的第一行是不变的，只要送显一次就可保持，不必重复送显；只有第二行是变化的。那么在应用中，对第一行只要送显一次就足够了，以后的任务是送显第二行的可变部分。所以，要求送显子程序中 PPU 的地址是可变的。只有这样才能使送显子程序的使用更灵活，通用性更强。

例 6—4 使用开关变址送显法送显图 6－1 的内容。

送显程序见 No. 6－4

No. 6－4 局部送显子程序(四)

```
7EF1   A9 FA      LDA  # $FA
7EF3   85 EE      STA  $EE
7EF5   A9 7E      LDA  # $7E
7EF7   85 EF      STA  $EF
7EF9   60         RTS
7EFA   20 00 7F   JSR  $7F00
7EFD   4C 0F 88   JMP  $880F
7F00   AD FF 04   LDA  $04FF    ;以 $4FF 为执行软开
                                关
7F03   D0 01      BN $  $7F06 ; $4FF≠0 执行送显
7F05   60         RTS ; $4FF＝0 则直接返回
7F06   8A         TXA
7F07   48         PHA
7F08   AD 30 7F   LDA  $7F30
```

```
7F0B   AE 31 7F    LDX $7F31
7F0E   20 F8 81    JSR $81F8 ;置 PPU 地址
7F11   A2 00       LDX #$00
7F13   BD 32 7F    LDA $7F32,X
7F16   F0 06       BEQ $7F1E
7F18   8D 07 20    STA $2007
7F1B   E8          INX
7F1C   D0 F5       BNE $7F13
7F1E   20 E6 AB    JSR $ABE6 ;写 PPU 后的屏幕定位
7F21   A9 00       LDA #$00
7F23   8D FF 04    STA $04FF ;关闭送显开关
7F26   68          PLA
7F27   AA          TAX
7F28   60          RTS
```

当程序 No. 6－4 装入中断处理程序后，运行 F BASIC 程序 No. 6－5 即可送显图 5－1 的局部画面。

```
5  REM No. 6－5 局部送显子程序(四)的辅助程序
10  POKE &7F30,20,01,54,49,4D,45,20,20,20,20,
    (PPU 首址 T I M E )
    53,43,4F,52,45,20,20,20,20,53,54,41,47,
    (S C O R E S T A G)
    45,20,20,20,4C,45,56,45,4C,00
    E L E V E L 送显结束标志)
20  POKE &4FF,1
30  POKE &7F30,20,21,30,38,3A,33,35,20,20,20,
    (PPU 首址 0 8 : 3 5 )
    30,31,32,33,34,20,20,20,20,20,20,33,20,
```

```
(0 1 2 3 4      3)
20,20,20,20,20,20,38,00
                    (8)
40    POKE &04FF,1
50    END
```

注——括号中的内容为注释，不要输入程序中。

显而易见，使用程序 No. 6－4、No. 6－5 可任意向屏幕的任意位置送显任意内容。

四、F BASIC 系统送显法

当局部送显的画面位于 F BASIC 的显示画面中时，还有更简单的编程方法。

例 6－5 在 F BASIC 画面座标(7, 5)处送显“WELCOME USING”，在(6,7)处送显“FAMILY COMPUTER”。实现程序见 No. 6－6

No. 6－6 局部送显子程序(五)

```
7F00 A9 00        LDA  # $00
7F02 85 44        STA  $44
7F04 A5 44        LDA  $44
7F06 C9 02        CMP  # $02
7F08 F0 07        BEQ  $7F11
7F0A 20 12 7F     JSR  $7F12 ;赋值子程序
7F0D E6 44        INC  $44
7F0F D0 F3        BNE  $7F04
7F11 60           RTS          ;送显结束返回
* * * *
7F12 0A           ASL
```

```
7F13 0A          ASL
7F14 AA          TAX
7F15 BD 00 7E    LDA $7E00,X
7F18 85 48       STA $48 ;取 X 座标
7F1A BD 01 7E    LDA $7E01,X
7F1D 85 49       STA $49 ;取 Y 座标
7F1F BD 02 7E    LDA $7E02,X
7F22 85 52       STA $52 ;取数据区首址
7F24 BD 03 7E    LDA $7E03,X
7F27 85 53       STA $53
7F29 4C F8 87    JMP $87F8 ;转读数送显子程序,送显
                           结束后返回
```

```
7E00 07 05 00 7D 06 07 0F 7D
座标 取数首址 座标 取数首址
7D00 57 45 4C 43 4F 4D 45 20 55 53 49 4E 47 00
     W E L C O M E  U S I N G
7D0F 46 41 4D 49 4C 59 20 20 43 4F 4D 50 55 54 45 52 00
     F A M I L Y  C O M P U T E R
```

程序 No. 6－6 由于调用了系统子程序，故短小精悍，玲珑剔透。显然，只要调整 $7E00 数据区就可随意控制显示位置和显示内容。

以上介绍的五个局部送显子程序中，前四个子程序都要写入中断程序才能运行，而子程序 No. 6－6 则可独立运行，这是它的突出特点。

五、F BASIC 程序快速送显法

在 BS·2A 版本 F BASIC 系统中，还可以不编机器语言

程序，直接编写 F BASIC 语言程序快速送显局部画面。

例 6－6 编写 F BASIC 语言程序快速送显背景图形库中第 32～254 号图形。

实现程序见 No. 6－7

```
5   REM 6－7 局部送显子程序(六)
10   CLS
20   FOR I=0 TO 223 : POKE 1792+I,I+32: NEXT
30   POKE 98,225
40   CALL 34811
```

程序 No. 6－7 中的 30 行写入 98 单元的数据为一次送显的数据个数加一，调整该数值可以控制一次送显的字符、图形数量。我们还可以在 40 行之前加一光标定位语句“LOCATE X,Y”用于控制送显内容的初始座标。由于该子程序调用了系统送显子程序 $8?FB，因而送显速度极快。有兴趣的朋友可以与用“PRINT CHR $”语句送显的方法比较一下程序运行速度，可以发现 No. 6－7 要快若干倍。

现在我们再使用 No. 6－7 试验送显例五的内容。程序见 No. 6－8

```
5   REM 6－8 局部送显子程序(七)
10    POKE 1792,&57,&45,&4C,&43,&4F,&4D,&45,
      &20,&55,&53,&45,&49,&4E,&47
20    POKE 98,15 : LOCATE 7,5: CALL 34811
30   POKE 1792,&46,&41,&4D,&49,&4C,&59,&20,&20,
      &43,&4F,&4D,&50,&55,&54,&45,&52
40    POKE 98,17 : LOCATE 6,7 :CALL 34811
```

运行 No. 6－8 可见，它达到了 No. 6－6 同样的效果。程序 No. 6－8 再一次体现了灵活调用系统子程序的优越性，同

时也表明，即使不编写机器语言程序也同样能加快程序的运行速度。

第二节　直接送显法全屏幕绘图

在背景画面的绘制中，常采取关显示后查表送显的方法。任天堂教学游戏《小学算术一年级》的绘图模式，就是查表法绘图的范例。

《小学算术一年级》游戏是日本东京小学推荐的优秀教学软件，它的背景画面绘制采取了最原始的方法——直接查表法。其绘图源程序如 No. 6－9。

No. 6－9《小学算术一年级》绘图源程序

```
AD04   8518     STA $18        ;保护 A 值
AD06   A900     LDA #$00
AD08   8516     STA $16        ;设置取数方式
AD0A   8619     STX $19        ;保护 X 值
AD0C   841A     STY $1A        ;保护 Y 值
AD0E   68       PLA
AD0F   850D     STA $0D
AD11   68       PLA
AD12   850E     STA $0E
AD14   A001     LDY #$01
AD16   B10D     LDA ($0D,Y)
AD18   8511     STA $11        ;绘图数据区地址低位
AD1A   C8       INY
AD1B   B10D     LDA ($0D,Y)
AD1D   8512     STA $12        ;绘图数据区地址高位
```

```
AD1F   A50D     LDA  $0D
AD21   18       CLC
AD22   6902     ADC  #$02
AD24   850D     STA  $0D
AD26   9002     BCC  $AD2A
AD28   E60E     INC  $0E
AD2A   A50E     LDA  $0E
AD2C   48       PHA
AD2D   A50D     LDA  $0D
AD2F   48       PHA            ;存子程序返回地址
AD30   A000     LDY  #$00
AD32   A516     LDA  $16
AD34   C902     CMP  #$02
AD36   F00A     BEQ  $AD42     ; 若$16=2 则不改变
                               PPU 地址
AD38   B111     LDA  ($11,Y)
AD3A   850F     STA  $0F       ;PPU 地址低位
AD3C   C8       INY
AD3D   B111     LDA  ($11,Y)
AD3F   8510     STA  $10       ;PPU 地址高位
AD41   C8       INY
AD42   B111     LDA  ($11,Y)
AD44   8514     STA  $14       ;每组送数的数量
AD46   C8       INY
AD47   B111     LDA  ($11,Y)
AD49   8515     STA  $15       ;本次送数组数
AD4B   C8       INY
```

```
AD4C   A514     LDA $14
AD4E   C515     CMP $15
AD50   9071     BCC $ADC3     ;若$14 < $15 则转
                              $ADC3,改为纵向送
                              数
AD52   A615     LDX $15
AD54   8A       TXA
AD55   48       PHA           ;送数组数入栈保护
AD56   A504     LDA $04
AD58   D032     BNE $AD8C
AD5A   A500     LDA $00
AD5C   48       PHA
AD5D   2035AC   JSR $AC35     ;屏蔽$2000的D7位
AD60   A500     LDA $00
AD62   8D0020   STA $2000     设置送数方式
AD65   A510     LDA $10
AD67   8D0620   STA $2006     ;置PPU地址高位
AD6A   A50F     LDA $0F
AD6C   8D0620   STA $2006     ;置PPU地址低位
AD6F   A614     LDX $14       ;以X为取数计数器
AD71   B111     LDA ($11,Y)
AD73   8D0720   STA $2007     ;  从数据区取数送入
                              PPU
AD76   A516     LDA $16
AD78   C901     CMP #$01      ;若$16=1则一个数据
                              连送X次
AD7A   F006     BEQ $AD82     ;若$16=0则连送其后
```

```
                                    的 X 个数
AD7C   E611     INC $11         ;令取数地址低位加一
AD7E   D002     BNE $AD82
AD80   E612     INC $12         ;若低位为零则令高位
                                  加一
AD82   CA       DEX
AD83   D0EC     BNE $AD71
AD85   68       PLA
AD86   2039AC   JSR $AC39
AD89   4CACAD   JMP $ADAC
AD8C   205DAE   JSR $AE5D
AD8F   A614     LDX $14
AD91   8A       TXA
AD92   2081AE   JSR $AE81
AD95   B111     LDA ($11,Y)
AD97   2081AE   JSR $AE81
AD9A   A516     LDA $16
AD9C   C901     CMP #$01
AD9E   F006     BEQ $ADA6
ADA0   E611     INC $11
ADA2   D002     BNE $ADA6
ADA4   E612     INC $12
ADA6   CA       DEX
ADA7   D0EC     BNE $AD95
ADA9   208DAE   JSR $AE8D
ADAC   A50F     LDA $0F         ;令 PPU 地址加 20H,
                                  使其指向下一行
```

```
ADAE   18      CLC
ADAF   6920    ADC #$20
ADB1   850F    STA $0F
ADB3   9002    BCC $ADB7
ADB5   E610    INC $10        ;若有进位则令高位加一
ADB7   68      PLA
ADB8   AA      TAX
ADB9   CA      DEX            ;令送数组数计数器减一
ADBA   D098    BNE $AD54      ;不为零则转$AD54继续送数
ADBC   A518    LDA $18        ;若为零则恢复原A、X、Y寄存器值,结束送数
ADBE   A619    LDX $19
ADC0   A41A    LDY $1A
ADC2   60      RTS
```

复位画面数据区如下：

B45E　　00 20

——

PPU 地址

B460 20 20 CE CE CE CE CE CE CE CE CE CE CE CE CE CE

└每组送数数量置于$14。以后的数据都是绘图数据

└本次送数的组数存于$15

```
B470 CE CE CE CE CE CE CE CE CE CE CE CE CE CE CE CE
B480 CE CE CE CE CE CE CE CE CE CE CE CE CE CE CE CE
B490 CE CE CE CE CE CE CE CE CE CE CE CE CE CE CE CE
B4A0 CE CE CE CE CE CE CE CE CE CE CE CE CE CE CE CE
B4B0 CE CE CE CE CE CE CE CE CE CE CE CE CE CE CE CE
B4C0 CE CE CE CE CE CE CE CE CE CE CE CE CE CE CE CE
B4D0 CE CE CE CE CE CE CE CE CE CE CE CE CE CE CE CE
B4E0 CE CE CE CE CE CE CE CE CE CE E0 E1 E2 E3 82 83
B4F0 82 83 CE CE CE A2 A3 CE CC CD CE CE CE CE CE CE
B500 CE CE CE CE CE CE CE CE CE CE E4 E5 E6 E7 92 93
B510 92 93 CE CE CE B2 B3 CE DC DD CE CE CE CE CE CE
B520 CE CE CE CE CE CE CE CE CE CE CE CE CE CE CE CE
B530 CE CE CE CE CE CE CE CE CE CE CE CE CE CE CE CE
B540 CE CE CE CE CE CE CE CE CE CE CE CE CE CE CE CE
B550 CE CE CE CE CE CE CE CE CE CE CE CE CE CE CE CE
B560 CE CE CE CE CE CE CE CE CE CE CE 80 81 80 81 E0
B570 E1 E2 E3 E8 E9 60 71 CC CD CE CE CE CE CE CE CE
B580 CE CE CE CE CE CE CE CE CE CE CE 90 91 90 91 E4
B590 E5 E6 E7 EA EB CE CE DC DD CE CE CE CE CE CE CE
B5A0 CE CE CE CE CE CE CE CE CE CE CE CE CE CE CE CE
B5B0 CE CE CE CE CE CE CE CE CE CE CE CE CE CE CE CE
B5C0 CE CE CE CE CE CE C4 C6 C6 C6 C6 C6 C6 C6 C6 C6
B5D0 C6 C6 C6 C6 C6 C6 C6 C6 C6 C6 C6 C6 C6 C5 CE CE
B5E0 CE CE CE CE CE CE C7 CF CF CF CF CF CF CF 8C 8D
B5F0 C8 CF E0 19 E2 E3 CF CF CF CF CF CF CF D7 CE CE
B600 CE CE CE CE CE CE C7 CF CF CF CF CF CF CF 9C 9D
B610 D8 D9 E4 E5 E6 E7 CF CF CF CF CF CF CF D7 CE CE
B620 CE CE CE CE CE CE C7 CF CF CF CF CF CF CF CF CF
B630 CF CF CF CF CF CF CF CF CF CF CF CF CF D7 CE CE
B640 CE CE CE CE CE CE C7 CF CF CF CF CF CF CF 84 85
B650 CA CB E0 19 E2 E3 CF CF CF CF CF CF CF D7 CE CE
B660 CE CE CE CE CE CE C7 CF CF CF CF CF CF CF 94 95
B670 DA DB E4 E5 E6 E7 CF CF CF CF CF CF CF D7 CE CE
B680 CE CE CE CE CE CE C7 CF CF CF CF CF CF CF CF CF
B690 CF CF CF CF CF CF CF CF CF CF CF CF CF D7 CE CE
B6A0 CE CE CE CE CE CE C7 CF CF CF CF CF CF CF 8C 8D
B6B0 C8 CF E0 19 E2 E3 CF CF CF CF CF CF CF D7 CE CE
B6C0 CE CE CE CE CE CE C7 CF CF CF CF CF CF CF 9C 9D
B6D0 D8 D9 E4 E5 E6 E7 CF CF CF CF CF CF CF D7 CE CE
B6E0 CE CE CE CE CE CE C7 CF CF CF CF CF CF CF CF CF
B6F0 CF CF CF CF CF CF CF CF CF CF CF CF CF D7 CE CE
B700 CE CE CE CE CE CE C7 CF CF CF CF CF CF CF 84 85
B710 CA CB E0 19 E2 E3 CF CF CF CF CF CF CF D7 CE CE
B720 CE CE CE CE CE CE C7 CF CF CF CF CF CF CF 94 95
B730 DA DB E4 E5 E6 E7 CF CF CF CF CF CF CF D7 CE CE
B740 CE CE CE CE CE CE D4 D6 D6 D6 D6 D6 D6 D6 D6 D6
```

```
B750  D6 D6 D6 D6 D6 D6 D6 D6 D6 D6 D6 D6 D6 D5 CE CE
B760  CE CE CF CF CF CF CF CF CF CF CF CF CF CF CF CF
B770  CF CF CF CF CF CF CF CF CF CF CF CF CF CF CF CF
B780  CF CF CF CF CF CF CF CF CF CF CF CF CF CF CF CF
B790  CF 08 18 09 08 CF CF 09 CF 18 CF CF CF CF CF CF
B7A0  CF CF CF CF CF CF CF CF CF CF CF CF CF CF CF CF
B7B0  CF CF CF CF CF CF CF CF CF CF CF CF CF CF CF CF
B7C0  CF CF CF CF CF CF CF CF CF CF CF CF CF CF CF CF
B7D0  CF CF CF CF CF CF CF CF CF CF CF CF CF CF CF CF
B7E0  CF CF CF CF CF CF CF CF CF CF CF CF CF CF CF CF
B7F0  CF CF CF CF CF CF CF CF CF CF CF CF CF CF CF CF
B800  CF CF CF CF CF CF CF CF CF CF CF CF CF CF CF CF
B810  CF CF CF CF CF CF CF CF CF CF CF CF CF CF CF CF
B820  CF CF 00 00 00 00 00 00 00 00 00 00 00 0B 02 08
B830  00 00 50 50 5E 52 50 5F 50 50 55 55 55 55 55 55
B840  55 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55
B850  56 56 00 00 00 00 00 00 00 00 50 50 50 50 50 50
B860
```

No. 6－9 所调用的子程序

```
AC35   A500      LDA  $00
AC37   297F      AND  #$7F
AC39   8D0020    STA  $2000
AC3C   8500      STA  $00
AC3E   60        RTS
```

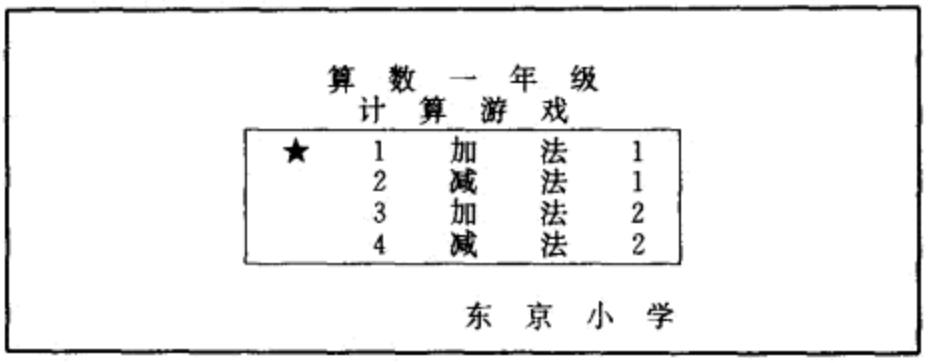
程序 No. 6－9 的绘图数据如图 6－2 所示。

列次	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F	行次
2000	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	00
2020	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	01
2040	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	02
2060	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	03
2080	CE	CE	CE	CE	CE	CE	CE	CE	E0	E1	E2	E3	82	83	82	83	CE	CE	CE	A2	A3	CE	CC	CD	CE	CE	CE	CE	CE	CE	CE	CE	04
20A0	CE	CE	CE	CE	CE	CE	CE	CE	E4	E5	E6	E7	92	93	92	93	CE	CE	CE	B2	B3	CE	DC	DD	CE	CE	CE	CE	CE	CE	CE	CE	05
20C0	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	06
20E0	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	07
2100	CE	CE	CE	CE	CE	CE	CE	CE	CE	80	81	80	81	E0	E1	E2	E3	E8	E9	60	71	CC	CD	CE	CE	CE	CE	CE	CE	CE	CE	CE	08
2120	CE	CE	CE	CE	CE	CE	CE	CE	CE	90	91	90	91	E4	E5	E6	E7	EA	EB	CE	CE	DC	DD	CE	CE	CE	CE	CE	CE	CE	CE	CE	09
2140	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	0A
2160	CE	CE	CE	CE	C4	C6	C6	C6	C6	C6	C6	C6	C6	C6	C6	C6	C6	C6	C6	C6	C6	C6	C6	C6	C6	C6	C6	C6	C5	CE	CE	CE	0B
2180	CE	CE	CE	CE	C7	•	•	•	•	•	•	•	8C	8D	C8	•	E0	19	E2	E3	•	•	•	•	•	•	•	D7	CE	CE	CE	CE	0C
21A0	CE	CE	CE	CE	C7	•	•	•	•	•	•	•	9C	9D	D8	D9	E4	E5	E6	E7	•	•	•	•	•	•	•	D7	CE	CE	CE	CE	0D
21C0	CE	CE	CE	CE	C7	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	D7	CE	CE	CE	CE	0E
21E0	CE	CE	CE	CE	C7	•	•	•	•	•	•	•	84	85	CA	CB	E0	19	E2	E3	•	•	•	•	•	•	•	D7	CE	CE	CE	CE	0F
2200	CE	CE	CE	CE	C7	•	•	•	•	•	•	•	94	95	DA	DB	E4	E5	E6	E7	•	•	•	•	•	•	•	D7	CE	CE	CE	CE	10
2220	CE	CE	CE	CE	C7	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	D7	CE	CE	CE	CE	11
2240	CE	CE	CE	CE	C7	•	•	•	•	•	•	•	8C	8D	C8	•	E0	19	E2	E3	•	•	•	•	•	•	•	D7	CE	CE	CE	CE	12
2260	CE	CE	CE	CE	C7	•	•	•	•	•	•	•	9C	9D	D8	D9	E4	E5	E6	E7	•	•	•	•	•	•	•	D7	CE	CE	CE	CE	13
2280	CE	CE	CE	CE	C7	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	D7	CE	CE	CE	CE	14
22A0	CE	CE	CE	CE	C7	•	•	•	•	•	•	•	84	85	CA	CB	E0	19	E2	E3	•	•	•	•	•	•	•	D7	CE	CE	CE	CE	15
22C0	CE	CE	CE	CE	C7	•	•	•	•	•	•	•	94	95	DA	DB	E4	E5	E6	E7	•	•	•	•	•	•	•	D7	CE	CE	CE	CE	16
22E0	CE	CE	CE	CE	D4	D6	D6	D6	D6	D6	D6	D6	D6	D6	D6	D6	D6	D6	D6	D6	D6	D6	D6	D6	D6	D6	D6	D5	CE	CE	CE	CE	17
2300	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	18
2320	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	08	18	09	08	•	•	09	•	18	•	•	•	•	•	•	•	19
2340	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	1A
2360	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	1B
2380	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	1C
23A0	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	1D
23C0	00	00	00	00	00	00	00	00	00	00	00	0B	02	08	00	00	50	50	5E	52	50	5E	50	50	55	55	55	55	55	55	55	55	
23E0	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	00	00	00	00	00	00	00	00	50	50	50	50	50	50	50	50	

注——"·"代表 CFH；¥23C0～¥23FF 两行为画面配色数据

图 6—2 《小学算术一年级》复位画面绘图数据

绘图效果如图 6－3 所示。



注——实际绘图效果是日文

图 6－3 《小学算术一年级》复位画面

程序 No. 6－9 中已作了详细注释，不多解释。下面给出供我们直接调用的子程序 No. 6－10。

No. 6－10（直接送显）背景绘图子程序（一）

```
7EF1  A9  FA     LDA # $FA
7EF3  85  EE     STA $EE
7EF5  A9  7E     LDA # $7E
7EF7  85  EF     STA $EF
7EF9  60  RTS
7EFA  20  00 7F  JSR $7F00
7EFD  4C  0F 88  JMP $880F
      * * * * *
7F00  48         PHA
7F01  8A         TXA
7F02  48         PHA
7F03  98         TYA
7F04  48         PHA
7F05  A9  00     LDA # $00
```

```
7F07  85  50     STA $50      ;绘图数据区地址低位存
                               $50
7F09  A9  78     LDA #$78
7F0B  85  51     STA $51      ;绘图数据区地址高位存
                               $51
7F0D  A0  00     LDY #$00
7F0F  B1  50     LDA ($50,Y)
7F11  85  52     STA $52      ;PPU 地址低位存$52
7F13  C8         INY
7F14  B1  50     LDA ($50,Y)
7F16  85  53     STA $53      ;PPU 地址高位存$53
7F18  C8         INY
7F19  B1  50     LDA ($50,Y)
7F1B  85  54     STA $54      ;每组送数的数量存$54
7F1D  C8         INY
7F1E  B1  50     LDA ($50,Y)
7F20  85  55     STA $55      ;本次送数组数存$55
7F22  C8         INY
7F24  A6  55     LDX $55
7F26  8A         TXA
7F27  48         PHA          ;送数组数入栈保护
7F28  A5  32     LDA $32
7F2A  48         PHA          ;$2000 原值入栈
7F2B  20  67 7F  JSR $7F67    ;屏蔽$2000 的 D7 位
7F2E  A5  32     LDA $32
7F30  8D  00 20  STA $2000    ;设置送数方式
7F33  A5  53     LDA $53
```

```
7F35  8D  06 20   STA  $2006    ;置 PPU 地址高位
7F38  A5  52      LDA  $52
7F3A  8D  06 20   STA  $2006    ;置 PPU 地址低位
7F3D  A6  54      LDX  $14      ;以 X 为取数计数器
7F3F  B1  50      LDA  ($50,Y)
7F41  8D  07 20   STA  $2007    ;从数据区取数送入 PPU
7F44  E6  50      INC  $50      ;令取数地址低位加一
7F46  D0  02      BNE  $7F4A
7F48  E6  51      INC  $51      ;若低位为零则令高位加一
7F4A  CA          DEX
7F4B  D0  F2      BNE  $7F3F
7F4D  68          PLA
7F4E  20  6B 7F   JSR  $7F6B
7F51  A5  52      LDA  $52      ;令 PPU 地址加 20H,
7F53  18          CLC           使其指向下一行
7F54  69  20      ADC  #$20
7F56  85  52      STA  $52
7F58  90  02      BCC  $7F5C
7F5A  E6  53      INC  $53      ;若有进位则令高位加一
7F5C  68          PLA
7F5D  AA          TAX
7F5E  CA          DEX           ;令送数组数计数器减一
7F5F  D0  C5      BNE  $7F26    ;不为零则转 $AD54 继续
                                送数
7F61  68          PLA           ;若为零则恢复原 A、X、Y
7F62  AA          TAX           寄存器值,结束送数
7F63  68          PHA
```

```
7F64  A8          TAY
7F65  68          PLA
7F66  60          RTS
      * * * * *
7F67  A5  32      LDA  $32
7F69  29  7F      AND  #$7F
7F6B  8D  00 20   STA  $2000
7F6E  85  32      STA  $32
7F70  60          RTS
```

现在，只要把绘图数据按 No. 6－9 中 $B45E 的格式写入 $7800，就可绘出你希望的背景画面。限于篇幅，不再举例，留给读者练习。

从 No. 6－9、No. 6－10 可明显看出，直接送显绘图法的优点是：绘图数据区为平面结构，数据与显示单元的关系一一对应，单纯从数据区就可看出绘图效果，背景画面设计直观、容易，修改方便，擅长于细腻画面的描绘，特别适合初学者使用；缺点是：由于每一个显示单元和配色单元都要占一个内存单元，一幅背景画面要占用 1024 个内存单元，故数据区较庞大。

第三节　BS・2A 背景绘图法

一、系统绘图子程序清单

BS・2A 版本的 F BASIC 工作系统的背景绘图子程序位于 ROM 区 $EE09～$EE56 单元。它是背景页绘图的一个重要程序，特复录绘图子程序于 No. 6－11。

N₀6－11 绘图子程序

```
EE09  8D  06 20  STA  $2006
EE0C  C8         INY
EE0D  B1  00     LDA  ( $00 ) , Y
EE0F  8D  06 20  STA  $2006
EE12  C8         INY
EE13  B1  00     LDA  ( $00 ) , Y
EE15  0A         ASL
EE16  48         PHA
EE17  A5  10     LDA  $10
EE19  09  04     ORA  # $04
EE1B  B0  02     BCS  $EE1F
EE1D  29  FB     AND  # $FB
EE1F  20  7A EC  JSR  $EC7A
EE22  68         PLA
EE23  0A         ASL
EE24  90  03     BCC  $EE29
EE26  09  02     ORA  # $02
EE28  C8 INY
EE29  4A         LSR
EE2A  4A         LSR
EE2B  AA         TAX
EE2C  B0  01     BCS  $EE2F
```

设置 PPU 首址

确定送数方式

```
EE2E C8          INY
EE2F B1  00      LDA ( $00 ) , Y
EE31 8D  07 20   STA $2007
EE34 CA          DEX
EE35 D0  F5      BNE $EE2C
EE37 38          SEC
EE38 98          TYA
EE39 65  00      ADC $00
EE3B 85  00      STA $00
EE3D A9  00      LDA # $00
EE3F 65  01      ADC $01
EE41 85  01      STA $01
EE43 AE  02 20   LDX $2002
EE46 A0  00      LDY # $00
EE48 B1  00      LDA ( $00 ) , Y
EE4A D0  BD      BNE $EE09
EE4C A5  12      LDA $12
EE4E 8D  05 20   STA $2005
EE51 A5  13      LDA $13
EE53 8D  05 20   STA $2005
EE56 60 RTS
```

向 PPU 送数

数据区地址指针
换算处理

向 PPU 送数的开始
和结束处理

二、复位画面的绘图数据

复位画面的绘图数据位于 ROM 区 $D7F4～$D9F9。

```
D7F4                  3F 00 20 0F 30 00 0F 0F 30 00 0F 0F
D800  30 00 0F 0F 30 0F 0F 0F 30 27 0F 0F 30 02 0F 0F
D810  30 16 0F 0F 30 2A 0F 23 C1 46 AA 23 C9 06 C0 F0
D820  F0 F0 F0 30 23 D1 06 CC FF FF FF FF 33 23 D9 06
D830  0C 0F 0F 0F 0F 03 23 E1 06 44 55 55 55 55 11 23
D840  E9 06 44 55 55 55 55 11 23 F0 50 AA 23 F1 46 55
D850  20 00 DE FE 20 01 DE 16 20 1E DE 02 20 1F DE FE
D860  20 BF CA 1E 22 3F C5 1E 20 02 DE 02 20 1D DE 16
D870  20 07 52 FE 20 27 52 FE 20 47 52 FE 20 06 C3 16
D880  20 19 C3 02 20 27 44 1C 20 2E 44 1C 20 35 44 1C
D890  20 47 44 1C 20 4E 44 1C 20 55 44 1C 20 A1 CA 01
D8A0  20 A2 CA 00 20 BD CA 01 20 BE CA 00 21 E1 02 1D
D8B0  1F 21 FD 02 1D 1F 20 C3 88 03 04 05 20 20 03 04
D8C0  05 20 E4 01 06 21 84 01 06 20 85 8C 09 0A 07 08
D8D0  07 07 07 07 08 07 0B 0C 20 86 54 0D 20 A6 54 0E
D8E0  21 C6 54 0F 21 E6 54 10 20 9A 8C 11 12 13 14 13
D8F0  13 13 13 14 13 15 18 20 FB 01 17 21 9B 01 17 20
D900  DC 88 19 1A 1B 20 20 19 1A 1B 22 21 C5 01 22 22
D910  C5 00 22 3D C5 01 22 3E C5 00 22 C1 02 1D 1F 22
D920  DD 02 1D 1F 22 43 83 03 04 05 22 64 01 06 22 05
D930  87 09 0A 07 08 07 0B 0C 22 5C 83 19 1A 1B 22 7B
D940  01 17 22 1A 87 11 12 13 14 13 15 18 22 06 54 0D
D950  22 26 54 0E 22 A6 54 0F 22 C6 54 10 23 02 C5 1C
```

D960 23 1D C5 1C 23 24 58 BC 23 64 58 BD 23 23 83 B9
D970 BA BB 23 29 83 B9 BA BB 23 2F 83 B9 BA BB 23 30
D980 83 B9 BA BB 23 36 83 B9 BA BB 23 3C 83 B9 BA BB
D990 23 05 82 BE BF 23 0B 82 BE BF 23 12 82 BE BF 23
D9A0 18 82 BE BF 00

下面划双线的为送数指南，划单线的为送入 PPU 的数据。

三、绘图原理分析

程序 No. 6－11 是一个结构最优化的查表法绘图子程序。虽然程序很短小，但却包含了 6527CPU 与 PPU 通信的全部操作。按功能可把它划分为两大部分：

$EE09～$EE41：向 PPU 送数处理程序。

$EE43～$EE56：对 PPU 进行开关操作处理程序。

下面结合背景页图形的绘制过程对程序 No. 6－11 分析如下：

1. PPU 复位

执行 $EE43 程序，读 $2002 软开关，使 PPU 复位。

2. 读数判断

根据 $00、$01 单元指示的数据区首址 $D7F4 取出第一个数据存入 A 寄存器，有 A＝3F。因 A≠0，转送数处理程序 $EE09。

3. 送数处理

送数处理程序由四部分组成。它们的工作过程如下：

(1) 设置 PPU 送数首址处理程序 $EE09～$EE11

这段程序的作用是从数据区中取出第一、第二两个数据

送入$2006，设置 PPU 的通信首址。

(2) 送数方式判断处理程序 $EE12～$EE28

这段程序决定向 PPU 的送数方式。它不仅能够决定一次向 PPU 送 a 个不同的数还是送 a 个相同的数，还能决定向 PPU 送数的取向，即是横向送数还是纵向送数。能够灵活调整向显示页面送数的走向，是 6527CPU 的又一大特点。后面将专题介绍送数方式的特点和工作过程。

(3) 送数程序 $EE29～$EE35

这段程序的作用是从数据区取数送入 PPU，并自动控制送数数量，达到送数数量后自动转出送数程序。

(4) 数据区地址指针换算程序 $EE37～$EE41

当一次送数完毕后，要及时调整 $00、$01 单元中的数据区指针，使其指向下一次送数的数据区首址。$EE37～$EE41 是专为换算数据区首址而设置的程序。

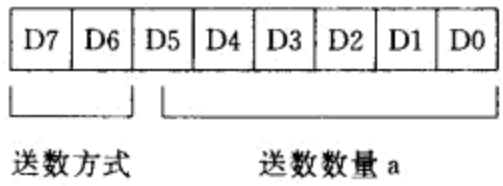
4. 送数方式讨论

送数处理程序 $EE09 全面体现了 6527CPU 的送数特点。它也是任天堂游戏中背景画面的重要绘制手段。现予以详细分析。

在绘图数据区 $D7F4～$DB2D 中，开始的三个数据就是核心数据，我们给它起一个名字叫送数指南。送数指南中的前两个数据决定向 PPU 置数的首址；第三个数据决定送数方式。现在重点讨论第三个数据。(注——在 PPU 数据区中，凡底下划横线的三个数据即为送数指南。为方便叙述，我们把这三个数据依次编号为 S1、S2、S3。)

(1) S3 各位的作用

S3 各位的作用如下图。



图中 D0～D5 位指示送数的数量 a，a 的最大值为 3FH，即一次最多可以送 3FH 个数据；D6、D7 位决定送数方式。因送数方式由两位二进制数表示，所以共有四种送数方式。送数方式代码和送数方式的对应关系见表 6－1。

表 6－1　PPU 送数方式和方式代码的对应表

方式代码		送　数　方　式
D7	D6	
0	0	横向送 S3 后 a 个单元的数据，每送一数 PPU 地址加 1 。
0	1	横向送 S3 后第一个单元的数据 a 次，每送一数 PPU 加 1。
1	0	纵向送 S3 后 a 个单元的数据，每送一数 PPU 地址加 H20。
1	1	纵向送 S3 后第一个单元的数据 a 次，每送一数 PPU 加 H20

(2) 下面结合程序的执行，讨论各种送数方式的实现。

·第一组送数指南地址在 $D7F4，数据为：

3F　　00　　20

S1　　S2　　S3

S1、S2 表示送数的 PPU 首地址为 $3F00，即 PPU 的配色区首址。S3＝20，其 D7、D6 位为 00，由表 6－1 可知，这意味着把 S3 后的 20H 个单元的数据送入 $3F00～$3F1F。以后，每送一数 PPU 地址自动加一，因此在执行 $EE12 以后的程序时，有

($10 OR 4)AND FB＝10→$2000

由于 $2000 的 D2 位等于 0，所以为横向送数方式。

因 A＝20，左移两次后，A＝80；仍有 C＝0，所以转

$EE29，令 A 右移两次，恢复 S3 原值，送入计数器 X；由 Y=Y+1 控制，把 $D7F7～$D816 的颜色代码依次送入 PPU 的 $3F00～$3F1F 中。

当 X=0 时，本次送数结束。通过数据区地址指针换算程序 $EE37～$EE41 的处理，令 $00=17、$01=D8，即数据区地址指向 $D817，进入下一轮取数判断、送数处理。

·第二组送数指南地址为 $D817，数据为：

23	C1	46
S1	S2	S3

置 PFU 首址为 $23C1。

现在 S3 的 D6=1，由表 6－1 可知这意味着对下一单元 $D81A 中的数据送 6 次。我们看一下在程序中是如何控制的。

执行 $EE12 以后的程序，有

A=$D819=46

A 左移一位后，A=8C、C=0，故仍有 $2000=10，置横向送数方式。A 左移一位后有 A=18、C=1，则有：

A=18 OR 2=1A

Y=Y+1=3

令 A 右移两次，返回原值。有 A=6、C=1，因而转 $EE2F，跳过了令 Y 加一的处理程序段，直接从 $D817+Y=D81A 中取数 AAH，送入 PPU。令 X－1 后，转 $EE2C，仍由于 C=1，跳过 Y 加一的处理，仍取 $D81A 的数据 AAH 送入 PPU。从而达到了令 S3 后第一个单元的数据连送 6 次的目的。向 $23C1～$23C6 中都置入了 AAH。

以后七组的送数情况与前两次相同(分析从略)。

·第十组送数指南的地址为 $D850，数据为：

20　　00　　DE

S1　　S2　　S3

显然，从本组开始，对背景 0 页的图形单元置数。因 S3＝DE，即 S3 的 D7、D6 位等于“11”。由表 6－1 可知这意味着向 PPU 纵向送 S3 后的第一个数据 FEH，连送 1EH 次，即占满背景 0 页的第一列。

下面我们分析程序的送数控制过程。

执行 $EE12、Y＝2、A＝$D850＋Y＝DE

A 右移一位后 A＝BC、C＝1

A 值入栈

令 A＝$10 OR 4＝14

现在，因 C＝1，故跳过 $EE1D，令 A 与 FBH 进行“与”运算后得 14H 送入 $2000。我们知道 $2000 开关组的 D2 位是纵向送数开关，现在该开关置 1（被打开），表示为纵向送数方式。

从堆栈弹出 A 值，A＝BC

A 左移一位有 A＝78、C＝1

因 C＝1 所以 A＝78 OR 2＝7A

令 Y＝Y＋1＝3

令 A 右移两位 A＝1E→X，C＝1

由于 C＝1 故转 $EE2F，跳过 Y＋1 的操作。

取 $D853 的数据 FEH 送入 PPU。然后，X 减一，CPU 自动控制 PPU 的地址加 20H，使下一 PPU 的送数地址为 $2020。转回 $EE2C。送下一个数……。直到 X＝0，本次送数结束。从而令背景 0 页的第一列 1EH 个单元全部置入 FEH。

·当送到第三十五组数据时，送数指南地址为 $D8B6，

送数指南为：

20	C3	88
S1	S2	S3

现在 S3 仅有 D7＝1，由表 6－1 可知这意味着从 PPU 的 $20C3 开始，连续纵向置入 S3 后的 8 个数据。

下面分析程序的控制过程：

执行 $EE12、Y＝2、A＝ $D8B6＋Y＝88

A 左移一位后 A＝10、C＝1

因 C＝1、所以 $2000＝14，置纵向送数方式。

A 左移一位，A＝20、C＝0

因 C＝0、转 $EE29、令 A 右移两位、A＝8、送入 X、C＝0

Y＝Y＋1＝3、A＝ $D8B6＋Y＝3→ $2007

X＝X－1＝7、转 $EE2C

Y＝Y＋1＝4、A＝ $D8B6＋Y＝4→ $2007

……

直至 X＝0，本次送数结束。从而把 S3 后的八个数据纵向送入了背景 0 页的八个单元。

以上分别讨论了程序 No. 6－11 对四种送数方式的控制过程。其它各组都包含在这四种方式之中，读者将很容易分析。

当读数 A＝ $D9A4＝0 时，则本次向 PPU 送数的操作结束。关闭 PPU，返回调用程序。

本次共向 PPU 送入了七十六组数据。第一组向 PPU 的配色单元置入颜色代码，其分布情况见表 3－6；第二～第九组向背景 0 页的配色单元送入配色代码，配色分布见图 3－2；第十～第七十六组向背景 0 页的图象单元送入了画面数据，绘出了复位画面中背景页的部分。画面数据分布图从略。

四、BS·2A 系统绘图法的应用

下面仅以《大金刚》一代游戏的第二关背景图形的绘制为例，介绍 BS·2A 版本 F BAISC 系统绘图子程序的应用。(因《大金刚》一代第二关的背景图形字模与我们的电脑系统的背景字模有些相似，故模拟比较容易。)

画面的绘图效果见《大金刚》一代第二关，在此仅将背景画面的设计中各显示单元的图形代号填入图 6－4(代号以丛书之二《游戏 BASIC 语言》中的背景图形库的标识为准)。

对照图 6－4，我们只要按程序 No. 6－11 的送数指南格式建立数据区后，再编写一个调用程序 N0. 6－11 的子程序就可以绘出《大金刚》一代第二关的游戏画面。画面数据如下：

```
7ED0  20 24 04 09 11 09 11 20 44 04 C4 1C 1C C6 20 63
7EE0  06 CF 0C BF BF 18 CF 20 83 06 E0 D8 C2 C2DA E1
7EF0  20 A4 04 1B C2 C2 04 20 C3 0A C9 BB 20 20 BB C9
7F00  20 D2 20 D2 20 0A C5 D2 20 0C C5 D2 20 8D 46 D0
7F10  20 AA 43 D0 20 B2 C2 D2 20 E1 10 D0 D0 D0 D0 C4
7F20  C5 C6 D0 D0 D0 D0 D0 D0 C4 C5 C6 21 F1 47 D0 21
7F30  25 03 C4 C5 C6 21 2E 03 C4 C5 C6 21 06 95 D1 C5
7F40  D3 D3 CF D3 D3 CF D3 D3 CF D3 D3 CF D3 D3 CF D3
7F50  D3 C5 D1 21 0F 95 D1 C5 D3 D3 CF D3 D3 CF D3 D3
7F60  CF D3 D3 CF D3 D3 CF D3 D3 C5 D1 22 02 43 D0 21
7F70  E9 44 D0 22 24 C5 D2 22 0A C7 D2 22 0C C7 D2 22
7F80  E3 C4 D2 21 17 C3 D2 21 5D C4 D2 21 B5 C3 D2 21
7F90  FB C3 D2 21 3D 42 D0 21 5A 42 D0 21 77 42 D0 21
7FA0  93 43 D0 21 DB 45 D0 22 15 42 D0 22 38 42 D0 22
```

列地址	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F
2000											H3		H3																			
2020					B2	C2	B2	C2			H3		H3																			
2040					F5	D5	D5	F7			H3		H3																			
2060				G8	B5	E8	E8	D1	G8		H3		H3																			
2080				J1	I1	F3	F3	I3	J2		H3		H3	H1	H1	H1	H1	H1	H1													
20A0					D4	F3	F3	A5			H1	H1	H1						H3													
20C0				G2	E4				G2		H3		H3						H3													
20E0		H1	H1	H1	H1	F5	F6	F7	H1	H1	H1	H1	H1	H1	F5	F6	F7	H1	H1	H1	H1	H1	H1	H1							
2100							H2									H2								H3							
2120						F5	F6	F7							F5	F6	F7							H3					H1	H1		
2140							H4									H4								H3				H1	H1		H3	
2160							H4									H4								H1	H1						H3	
2180							G8									G8				H1	H1	H1								H3		
21A0							H4									H4						H3								H3		
21C0							H4									H4						H3						H1	H1	H1	H1	H1
21E0							G8			H1	H1	H1	H1			G8						H3						H3				
2200			H1	H1	H1		H4				H3		H3			H4						H1	H1					H3				
2220					H3		H4				H3		H3			H4									H1	H1		H3				
2240					H3		G8				H3					G8												H1	H1			
2260					H3		H4				H3					H4														H1	H1	
2280					H3		H4				H3					H4														H3		
22A0					H3		G8				H3					G8														H1	H1	H1
22C0			H1	H1	H1		H4				H3					H4																
22E0				H3			H4				H1					H4										H1	H1					
2300				H3			G8									G8																
2320				H3			H4									H4								H1	H1							
2340				H3			H4									H4																
2360			H1	H1	H1	F5	F6	F7							F5	F6	F7			H1	H1	H1										
2380							H2									H2																
23A0	H1	H1	H1	H1	H1	G1	G2	G3	H1	H1	H1	H1	H1	H1	G1	G3	G2	H1	H1	H1	H1	H1	H1	H1	H1	H1	H1	H1	H1	H1	H1	H1

图 6—4

```
7FB0  5C 42 D0 22 7E 42 D0 22 9E 01 D2 22 BD 43 D0 22
7FC0  C2 43 D0 22 EA 43 D0 22 FA 42 D0 23 38 42 D0 23
7FD0  62 06 D0 D0 D0 C4 C5 C6 23 6E 08 C4 C5 C6 20 20
7FE0  D0 D0 D0 23 A0 45 D0 23 A5 03 C8 CA C9 23 A8 46
7FF0  D0 23 AE 03 C8 CA C9 23 B1 4F D0 00
```

以上数据区仅占用了300个内存单元，比直接送显绘图法要节约三分之二的数据区，这是它的突出优势。

使用BS·2A版本F BASIC的朋友可直接调用系统送显子程序，调用程序见No.6－12。

No.6－12（BS.2A绘图子程序$EE43调用子程序）

背景绘图子程序(二)

```
7EA0 48          PHA
7EA1 8A          TXA
7EA2 48          PHA
7EA3 98          TYA
7EA4 48          PHA
7EA5 A5 00       LDA  $00
7EA7 48          PHA
7EA8 A5 01       LDA  $01
7EAA 48          PHA              ;保护现场
7EAB A9 00       LDA  #$00
7EAD 8D 01 20    STA  $2001       ;关显示
7EB0 A9 D0       LDA  #$D0
7EB2 85 00       STA  $00
7EB4 A9 7E       LDA  #$7E
7EB6 85 01       STA  $01         ;向$00、$01置入数据
```

```
                                    区首址
7EB8 20 43 EE   JSR $EE43       ;系统送显子程序
7EBB A9 1E      LDA #$1E
7EBD 8D 01 20   STA $2001       ;开显示
7EC0 68         PLA
7EC1 85 01      STA $01
7EC3 68         PLA
7EC4 85 00      STA $00
7EC6 68         PLA
7EC7 A8         TAY
7EC8 68         PLA
7EC9 AA         TAX
7ECA 68         PLA
7ECB 60         RTS
```

当欲在其它 F BASIC 版本下运行 No. 6－12 子程序时，则要将 No. 6－11 合成到 No. 6－12 中才能绘出背景图案。程序的合成留给读者练习。

第四节　置色绘图法

任天堂游戏《大赛车》复位画面的绘制方法与 BS・2A 的背景绘图法截然不同，它的标题画面大黑体美术字

F－1

RACE

的绘制，采取了极为奇特的绘图方法，使程序量、数据区都得到了极大的压缩。现介绍给大家，《大赛车》的绘图源程序见 No. 6－13。

一、《大赛车》复位画面绘图子程序清单

No. 6—13《大赛车》复位画面绘图子程序

```
E62B A920     LDA # $20
E62D A040     LDY # $40
E62F 2064E9   JSR $E964      ;置 PPU 送数首址 $2040
E632 A000     LDY # $00
E634 B96DE8   LDA $E86D,Y
E637 F022     BEQ $E65B
E639 C920     CMP # $20
E63B B018     BCS $E655
E63D 290F     AND # $0F
E63F AA       TAX
E640 B96DE8   LDA $E86D,Y
E643 2910     AND # $10
E645 D004     BNE $E64B
E647 A92D     LDA # $2D
E649 D002     BNE $E64D
E64B A92E     LDA # $2E
E64D 8D0720   STA $2007
E650 CA       DEX
E651 10FA     BPL $E64D
E653 3003     BMI $E658
E655 8D0720   STA $2007
E658 C8       INY
E659 D0D9     BNE $E634
E65B 60       RTS
```

绘图数据区

```
E86D                                                    0A 13 FE
E870    02 A9 11 09 2F 2D 2F 2D 2F 2D 2F 2D 2F 01 11 05
E880    12 01 2F 2D 2F 2D 2F 2D 2F 2D 2D 2F 2D 2F 2D 2F
E890    2D 2F 02 12 FE 2D 11 01 11 02 2F 2D 2F 2D 2F 2D
E8A0    2F 2F 2D 2F 2D 2F 2D 2F 03 11 06 11 03 2F 2D 2F
E8B0    2D 2F 2D 2D 2F 2D 2F 2D 2F 04 11 06 11 04 2F 2D
E8C0    2F 2D 2F 0F 0F 0F 0F 04 13 FB 2D FA 12 FB 2D FA
E8D0    12 FB 2D 13 FE 08 11 2D 11 2D 11 2D 11 2D 11 2D
E8E0    11 2D 11 0B 13 FD 2D 11 2D 11 2D 11 03 12 FE 09
E8F0    11 FF 2E BA 2D 14 2D 11 2D 11 2D 11 0B 11 2D FF
E900    2E 2D 11 2D 11 2D FC 12 FD 2D 13 FE 03 00 00 00
```

程序 No. 6－13 中，若从数据区读出的数据为零则结束送数；若读出的数据大于或等于 20H 则将该数直接送入 PPU；若读出的数据小于 20H，且该数的 D4 位为零则以该数为计数器连向 PPU 送 2DH(《大赛车》游戏中背景图形库的 2DH 号图形为空格)；若读出的数据小于 20H，且其 D4 位为 1，则以该数的低四位为计数器连向 PPU 送 2DH。程序 No. 6－13 仅占用数据区 160 个单元，就完成了向 PPU 的 $2040～$21BF 共 384 个单元置入数据的任务。

二、《大赛车》复位画面的绘图效果

《大赛车》复位画面的绘图效果如图 6－5 所示，有关画面数据的填写留给读者练习。

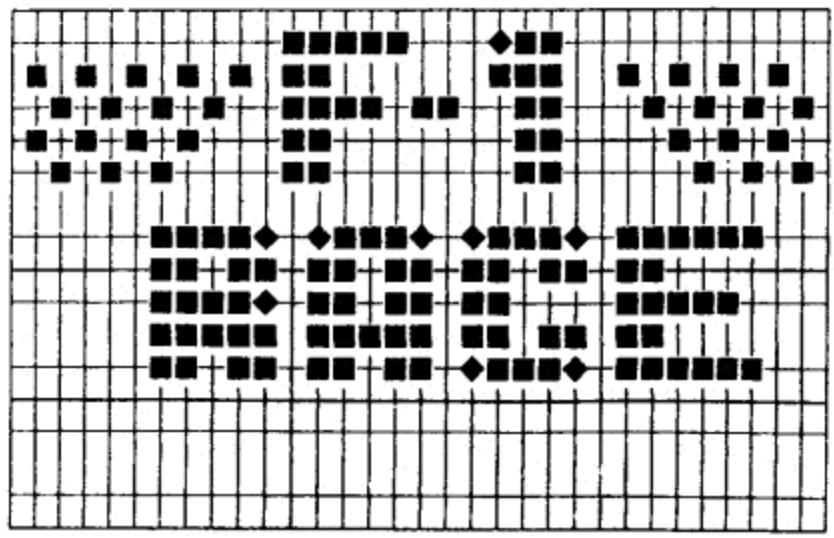


图 6－5 《大赛车》标题画面

三、置色绘图法的程序设计

下面介绍置色绘图法的程序设计过程。

我们知道，背景画面中每相连的十六个显示元素使用一个配色单元，那么，使用置色法绘图可使绘图数据区至少压缩十六倍。因此，在大面积简单图形的绘制中，这是一种最优化的绘图方法。下面通过例六详细介绍这种绘图方法的应用。

例 6－6 在背景 00 页、10 页上显示如图 6－6 的四个巨型汉字“游艺之窗”，并令两页画面循环移位显示。

1. 编程思路

首先定义一个配色代码为全黑色，如选定 0＃配色；然后把两背景页的显示映射区全部填满选定的图形，如选定代码为 255 的光标块；最后把欲显示的图形的配色单元置入黑色，其它置为可见色，也可以反过来把图形部分置为可见色，非图形部分置为黑色。置数操作由机器语言程序完成，画面

位移用 F BASIC 程序控制。

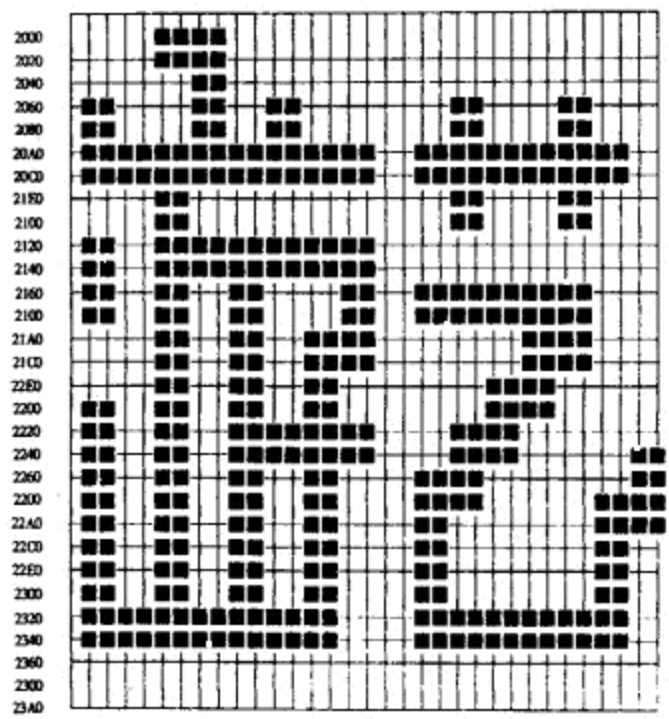


图 6- 6(A)　背景 00 页

2. 编程步骤

把“游艺之窗”四字分别复制到 00、10 两页的屏幕映射单元的座标图中，见图 6—7、图 6—8。

(2)将图形部分全部填入“00”号配色代码，空白部分填入“11”号配色代码(为使画面更华美，本例的空白区填入了“10、11”两种配色代码。)。

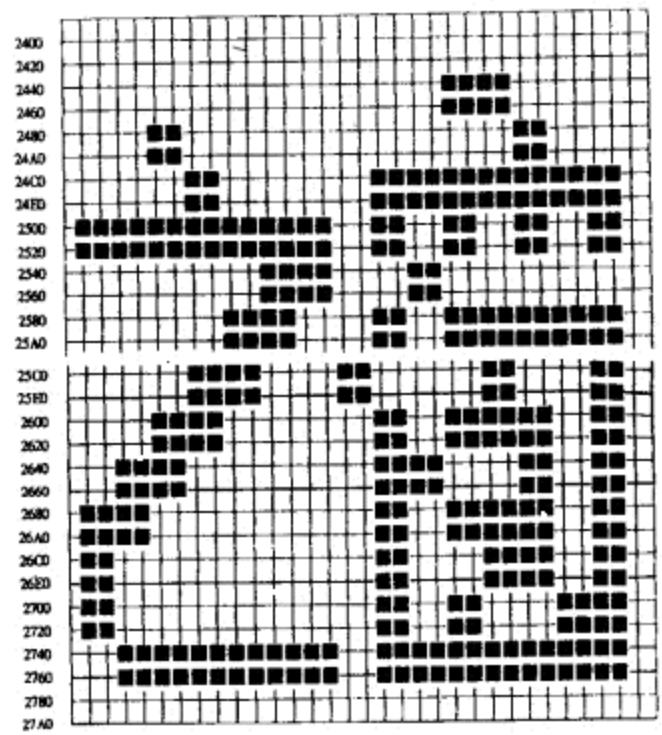


图 6　6(B)　背景 10 页

(3)根据各配色单元的配色代码,组合出两显示页面的配色数据。

背景 00 页的配色数据如下:

23C0 FF 30 FF FF FF FF FF FF 0C 03 03 0F 3F 0C 03 CF

23D0 CF 0C 0F 0F FF FC FC FF FC CC CC 03 F3 F0 00 FF

23E0 8A 88 08 08 AA 02 A8 AA 88 88 88 88 22 A8 AA 8A

23F0 08 28 28 A8 22 0A 0A 88 0A 0A 0A 0A 0A 0A 0A 0A

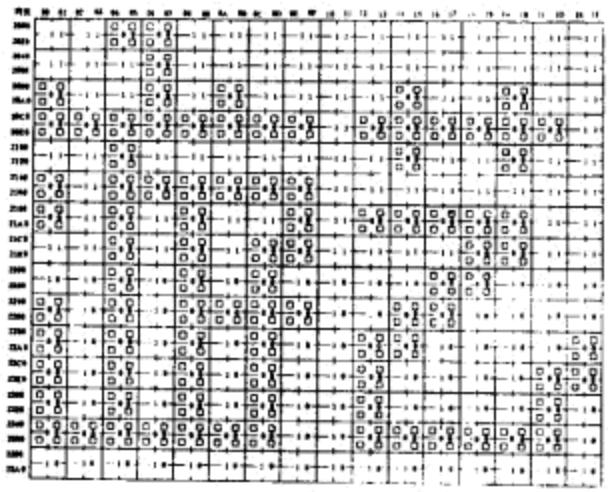


图 6－7　背景 00 页中“游艺”两字的配色坐标图

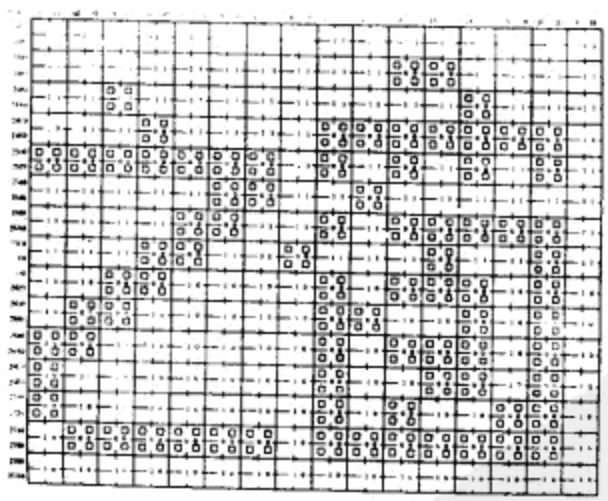


图 6－8　背景 10 页中“之窗”两字的配色坐标图

背景 10 页的配色数据如下：

27C0 FF FF FF FF FF 0F FF FF FF 3C FF FF 0F 0F 0C CF
27D0 F0 F0 30 CC 3C FC FC FC FF 3F C0 3F FC 30 F0 CC
27E0 2A 80 AA AA 08 A0 88 88 80 AA AA AA 88 20 88 88
27F0 28 0A 0A 8A 08 08 02 88 0A 0A 0A 0A 0A 0A 0A 0A

(4)确定 PPU 的配色。我们选取 0#配色代码为黑色、1#配色代码为蓝色、2#配色代码为绿色、3#配色代码为红色，则 PPU 的配色区的颜色代码如下：

3F00 0F 0F 0F 0F 0F 02 02 02 0F 0A 0A 0A 0F 05 05 05

(5)确定程序区、数据区地址。由于程序量和数据量都不太大，故我们选定 $7E00 为程序区首址、$7F00 为数据区首址。把 PPU 配色数据写入 $7F00～$7F0F、背景 00 页的配色数据写入 $7F10～$7F4F、背景 10 页的配色数据写入 $7F50～$7F8F。

(6)编写程序。为便于理解，采取顺序结构编写，读者可把它改写成其它结构形式。机器语言程序见 No. 6－14。

No. 6－14 (置色绘图法)背景绘图子程序(三)

```
7E00 48       PHA
7E01 8A       TXA
7E02 48       PHA
7E03 98       TYA
7E04 48       PHA
7E05 A0 00    LDY #$00
7E07 8C 01 20 STY $2001
7E0A A2 20    LDX #$20
7E0C 8E 06 20 STX $2006
7E0F 8C 06 20 STY $2006
```

```
7E12 A0 FF      LDY # $FF
7E14 A9 FF      LDA # $FF
7E16 8D 07 20   STA $2007
7E19 88         DEY
7E1A D0 FA      BNE $7E16
7E1C E8         INX
7E1D E0 28      CPX # $28
7E1F D0 EB      BNE $7E0C ;将两背景页全部置入 FFH
7E21 A9 3F      LDA # $3F
7E23 A0 00      LDY # $00
7E25 8D 06 20   STA $2006
7E28 8C 06 20   STY $2006
7E2B B9 00 7F   LDA $7F00,Y
7E2E 8D 07 20   STA $2007
7E31 C8         INY
7E32 C0 10      CPY # $10
7E34 D0 F5      BNE $7E2B ;置 PPU 配色
7E36 A9 23      LDA # $23
7E38 8D 06 20   STA $2006
7E3B A9 C0      LDA # $C0
7E3D 8D 06 20   STA $2006
7E40 B9 00 7F   LDA $7F00,Y
7E43 8D 07 20   STA $2007
7E46 C8         INY
7E47 C0 50      CPY # $50
7E49 D0 F5      BNE $7E40 ;送 00 页配色
7E4B A9 27      LDA # $27
```

```
7E4D 8D 06 20   STA $2006
7E50 A9 C0      LDA #$C0
7E52 8D 06 20   STA $2006
7E55 B9 00 7F   LDA $7F00,Y
7E58 8D 07 20   STA $2007
7E5B C8         INY
7E5C C0 90      CPY #$90
7E5E D0 F5      BNE $7E55 ;送 10 页配色
7E60 A5 33      LDA $33
7E62 8D 01 20   STA $2001
7E65 68         PLA
7E66 A8         TAY
7E67 68         PLA
7E68 AA         TAX
7E69 68         PLA
7E6A 60         RTS
```

(7)编写 No. 6－14 的写入程序和画面位移控制程序。见 No. 6－15。

```
5 REM No. 6－15 置色绘图子程序
10 FOR I=0 TO 106 : READ A$ : POKE &7E00+I ,
   VAL("&"+A$) : NEXT
20 FOR I=0 TO 143 : READ A$ : POKE &7F00+I ,
   VAL("&"+A$) : NEXT
30 DATA 48,8A,48,98,48,A0,00,8C,01,20,A2,
   20,8E,06, 20, 8C,06,20,A0,FF,A9,FF,8D,
   07,20,88,D0,FA,E8,E0,28, D0,EB,A9,3F,
   A0,00,8D,06,20,8C,06,20,B9,00,7F,8D,
```

```
    07,20,C8,C0,10,D0,F5
40 DATA A9,23,8D,06,20,A9,C0,8D,06,20,B9,
    00,7F,8D,07, 20,C8,C0,50,D0,F5,A9,27,
    8D,06,20,A9,C0,8D,06,20, B9,00,7F,8D,
    07,20,C8,C0,90,D0,F5,A5,33,8D,01,20,
    68,A8,68,AA,68,60
50 DATA 0F,0F,0F,0F,0F,02,02,02,0F,0A,0A, 0A,0F,
    05, 05, 05,FF,30,FF,FF,FF,FF,FF, FF,0C,03,03,
    0F,3F,0C, 03, CF,CF,0C,0F, 0F,FF,FC,FC,FF,FC,
    CC,CC,03,F3,F0,00,
60 DATA FF,8A,88,08,08,AA,02,A8,AA,88,88,
    88,88,22, A8, AA,8A,08,28,28,A8,22,0A,
    0A,88,0A,0A,0A,0A,0A, 0A, 0A,0A,FF,FF,
    FF,FF,FF,0F,FF,FF,FF,3C,FF,FF,0F,0F,
    0C,CF,F0,F0,30,CC,3C,FC,FC,FC
70 DATA FF,3F,C0,3F,FC,30,F0,CC,2A,80,AA, AA,
    08,A0,88,88,80,AA,AA,AA,88,20,88, 88,28,0A,
    0A,8A,08,08,02, 88,0A,0A,0A, 0A,0A,0A,0A,0A
80 CALL &7E00
90 A=0
100 POKE 8197,A : POKE 8197,0
110 A=A+1 : IF A>255 THEN 140
120 PAUSE 5
130 GOTO 100
140 IF B=0 THEN POKE 8192,144 : B=1 : GOTO 90
150 POKE 8192,145 : B=0 : GOTO 90
160 END
```

输入程序 No. 6－15，检查无误后，运行程序，将看到“游艺之窗”四个巨型大字缓缓左移，周而复始，十分迷人。

程序 No. 6－15 仅仅使用了 144 个数据就完成了两幅画面的绘制，其绘图效率是极高的。若再调用系统子程序并使用双重循环，程序量还可进一步压缩。可见，在大面积简单图形的绘制中，置色绘图法是最优化的一种。

第五节　优化查表法绘图

《魂斗罗》游戏已为广大朋友所熟悉、所喜爱。该游戏的编程中有许多优化的方法可供我们学习和借鉴。兹特列两例供参考。

一、武器选项画面的绘制

压缩版的《魂斗罗》游戏开始有一个武器选项画面（见图 6－9）。在该画面，游戏者可通过选择键（SELECT）在四种武器中任选其一。该画面的绘制程序见 No. 6－16。

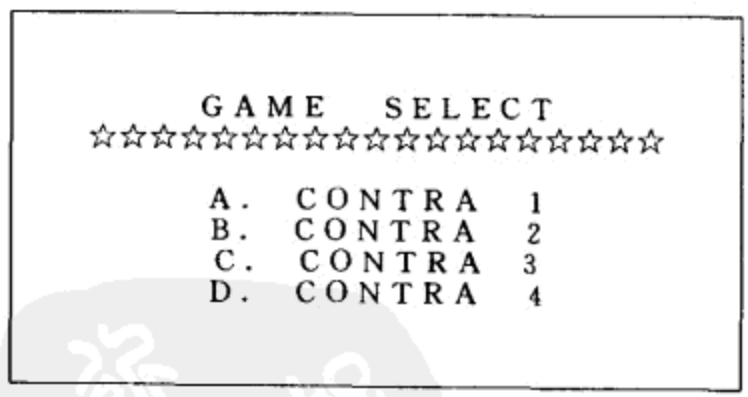


图 6－9　《魂斗罗》武器选项画面

No. 6－16《魂斗罗》武器选项画面的绘制程序

```
F80D A94E      LDA #$4E
```

```
F80F 8500      STA $00
F811 A9F8      LDA #$F8
F813 8501      STA $01          ;置绘图数据区首址于
                                  $00、$01 单元
F815 A000      LDY #$00
F817 B100      LDA ($00,Y)
F819 8D0620    STA $2006        ;置 PPU 地址高位
F81C C8        INY
F81D B100      LDA ($00,Y)
F81F 8D0620    STA $2006        ;置 PPU 地址低位
F822 C8        INY
F823 B100      LDA ($00,Y)
F825 AA        TAX
F826 3011      BMI $F839        ;判断送数指南,若大于
                                  7FH 则转 $F839
F828 C8        INY
F829 B100      LDA ($00,Y)      ;送数指南小于 80H 的送
                                  数处理,将其后的 N 个
                                  数送入 PPU 。
F82B 8D0720    STA $2007
F82E CA        DEX
F82F D0F7      BNE $F828
F831 C8        INY
F832 B100      LDA ($00,Y)
F834 C980      CMP #$80
F836 D0DF      BNE $F817        ;不是 80H 改变 PPU 地
                                  址,继续送数
```

```
F838 60      RTS              ;如是 80H 则结束送数
F839 8A      TXA
F83A 297F    AND # $ 7F
F83C AA      TAX              ;以该数的 D6～D0 位为计
                                数器,将其后的一数连送
                                N 次。
F83D C8      INY
F83E B100    LDA ($ 00,Y)
F840 8D0720  STA $ 2007
F843 CA      DEX
F844 D0FA    BNE $ F840
F846 F0E9    BEQ $ F831
```

数据区如下:

```
F84E                                                    3F 00
                                                        PPU配
F850  08 0F 20 20 20 0F 27 27 27 3F 10 04 0F 05 0F 17
      色,送 $ 3F00～$ 3F07。  送 PPU $ 3F10～$ 3F13
F860  23 D0 88 50 20 E4 17 20 20 20 20 20 20 20 20 20
      背景零页配色,$ 23D0～$ 23D7
F870  20 20 20 20 20 20 20 20 20 20 20 20 20 20 21 2A
F880  0B 47 41 4D 45 00 53 45 4C 45 43 54 21 66 93 3F
        G A M E  S E L E C T
F890  21 AA 0B 41 40 00 43 4F 4E 54 52 41 00 31 21 EA
            A . C O N T R A 1
F8A0  42 40 00 43 4F 4E 54 52 41 00 32 22 2A 0B 43
          B . C O N T R A 2 C
```

```
F8B0  40 00 43 4F 4E 54 52 41 00 33 22 6A 0B 44 40 00
      .  C O N T R A  3  D .
F8C0  43 4F 4E 54 52 41 00 34 00 00
      C O N T R A  4
```

图6－9的数据区仅使用了124个单元就完成了PPU配色、背景页配色和画面显示，虽然四行都是相同的内容(CONTRA)，但数据区中已留出了相应的空间，即使送显其它内容，数据区也不会增加。

程序No. 6－16的结构比较简单，读者可自己完成程序的转换，不多赘述。

二、《魂斗罗》标题画面的绘制

我们知道，《魂斗罗》的标题画面是由画写的“CONTRA”和两个主人翁的上身组成的。画面既简单抽象又富有力的感召，是正义和美的有机结合与升华。《魂斗罗》仅凭一幅标题画面就勾起了人们的无限遐思与向往，这怎能不令人钦佩和惊叹呢！现在让我们看看这一标题画面是如何绘制的。

标题画面的绘制程序在$C9CF～$CA60，其数据区在第二体的$9097～$9251，现抄录如下，见程序No. 6－17。

No. 5－26《魂斗罗》标题画面绘制程序

```
C9CF A001     LDY #$01      ;调用前已向$00、$01
C9D1 B100     LDA ($00,Y)     单元置入数据区首址
$9097
C9D3 8D0620   STA $2006     ;置PPU地址高位
C9D6 88       DEY
```

```
C9D7 B100     LDA ($00,Y)
C9D9 8D0620   STA $2006     ;置 PPU 地址低位
C9DC A902     LDA #$02
C9DE A604     LDX $04
C9E0 1001     BPL $C9E3
C9E2 0A       ASL           ;若 $04>7F 则 A=4
C9E3 A200     LDX #$00
C9E5 2092C8   JSR $C892     ;调整取数地址。
C9E8 A000     LDY #$00
C9EA B100     LDA ($00,Y)
C9EC C9FF     CMP #$FF      ;若 A=FF 则结束送数
C9EE F06B     BEQ $CA5B
C9F0 C97F     CMP #$7F      ;若 A=7F 则改变 PPU 地
                            址
C9F2 F05D     BEQ $CA51
C9F4 A8       TAY
C9F5 1026     BPL $CA1D
C9F7 297F     AND #$7F
C9F9 8502     STA $02       ;若取数指南小于7FH 则
C9FB A001     LDY #$01      以$02为计数器,送后面
C9FD B100     LDA ($00,Y)   的 N 个数。
C9FF A604     LDX $04
CA01 1003     BPL $CA06
CA03 2036CA   JSR $CA36     ;若 $04>7F 则延时送数
CA06 8D0720   STA $2007
CA09 C402     CPY $02
CA0B F003     BEQ $CA10     ;  达到送数数量则转
```

```
                                    $CA10调整地址
CA0D C8       INY                   ;未达到送数数量则转
CA0E D0ED     BNE $C9FD             $C9FD 继续送数
CA10 A901     LDA #$01
CA12 18       CLC
CA13 6502     ADC $02
CA15 A200     LDX #$00
CA17 2092C8   JSR $C892             ;调整取数地址
CA1A 4CE8C9   JMP $C9E8
CA1D A001     LDY #$01              ;当取数指南>7FH 时,把
CA1F 8502     STA $02               后面的一个数据连送 N 次
                                    的处理。
CA21 B100     LDA ($00,Y)
CA23 A402     LDY $02
CA25 A604     LDX $04
CA27 1003     BPL $CA2C
CA29 2036CA   JSR $CA36
CA2C 8D0720   STA $2007
CA2F 88       DEY
CA30 D0FA     BNE $CA2C
CA32 A902     LDA #$02
CA34 D0DF     BNE $CA15             ;一次送数结束,调整地址
CA36 8503     STA $03               ;以下是延时子程序
CA38 0603     ASL $03
CA3A 6A       ROR
CA3B 0603     ASL $03
CA3D 6A       ROR
```

```
CA3E 0603     ASL  $03
CA40 6A       ROR
CA41 0603     ASL  $03
CA43 6A       ROR
CA44 0603     ASL  $03
CA46 6A       ROR
CA47 0603     ASL  $03
CA49 6A       ROR
CA4A 0603     ASL  $03
CA4C 6A       ROR
CA4D 0603     ASL  $03
CA4F 6A       ROR
CA50 60       RTS
CA51 A901     LDA  #$01
CA53 A200     LDX  #$00
CA55 2092C8   JSR  $C892      ;调整取数地址
CA58 4CCFC9   JMP  $C9CF
CA5B 60       RTS             ;结束送数
```

调整取数地址子程序

```
C892 18       CLC
C893 7500     ADC  $00,X
C895 9500     STA  $00,X
C897 9002     BCC  $C89B
C899 F601     INC  $01,X
C89B 60       RTS
```

绘图数据

```
9097                          00 20 4B 00 8B 00 00 00
```

PPU $2000 先送4BH 个00、又送

90A0 00 00 00 00 00 00 00 14 00 8C 00 00 00 00 00 00

BH 个00　　送14H 个00，送0CH 个00（下同）

90B0 00 00 00 00 00 00 14 00 82 00 00 3A 00 86 02 03

90C0 04 05 06 07 19 00 89 11 12 13 14 15 16 17 18 19

90D0 16 00 8A 20 21 22 23 24 25 26 27 28 29 16 00 83

90E0 60 61 62 04 00 82 08 09 17 00 96 70 71 72 00 00

90F0 63 64 65 66 67 68 69 6A 6B 6C 6D 6E 6F 0A 0B 0C

9100 0D 0A 00 97 80 81 82 00 00 73 74 75 76 77 78 79

9110 7A 7B 7C 7D 7E 7F 1A 1B 1C 1D 1E 09 00 98 90 91

9120 92 00 00 83 84 85 86 87 88 89 8A 00 8C 8D 8E 8F

9130 2A 2B 2C 2D 2E 2F 08 00 98 A0 A1 A2 A3 00 93 94

9140 95 96 97 98 99 9A 00 9C 9D 9E 9F 3A 3B 3C 3D 3E

9150 3F 09 00 8B B1 B2 B3 B4 B5 B6 B7 B8 B9 BA BB 17

9160 00 86 A4 A5 A6 A7 A8 A9 27 00 85 C4 00 D0 D1 D2

9170 1A 00 87 D3 D4 D5 E0 E1 E2 E3 0B 00 8B 50 4C 41

P L A

9180 59 00 53 45 4C 45 43 54 04 00 85 E4 E5 F0 F1 F2

Y S E L E C T

9190 1A 00 88 F3 F4 F5 C5 C6 C7 C8 C9 0C 00 88 31 00

1

91A0 50 4C 41 59 45 52 03 00 85 CA CB CC CD CE 03 00

P L A Y E R

91B0 82 EF D6 15 00 86 D9 DA DB DC DD DE 04 00 81 E6

91C0 0B 00 90 32 00 50 4C 41 59 45 52 53 00 E9 EA EB

2 P L A Y E R S

91D0 EC ED EE 04 00 81 F6 17 00 84 D7 D8 E7 E8 13 00

91E0 8D 00 00 00 00 00 00 00 00 00 00 00 00 00 0E 00

91F0 97 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

```
9200  00 00 00 00 00 00 00 00 10 00 8B 00 00 00 00 00
9210  00 00 00 00 00 00 0E 00 97 00 00 00 00 00 00 00
9220  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
      以下为配色数据
9230  4D 00 83 A0 A0 20 06 00 82 58 52 03 50 83 00 00
9240  20 05 05 05 00 83 CC FF F3 05 00 83 FF FF CC 11
9250  00 FF
```

注——数据区中，凡下面划双线的数据为取数指南。

程序 No. 6－17使用了442个数据完成了标题画面的绘制。实际上单对标题画面而言，数据量还可以压缩。如在清屏以后绘图时，数据区中的 $9097～$90BC 和 $91E0～$922F 这118个置零(清屏)的数据可以删除，那么数据区将减少到326个单元。可见，程序 No. 6－17是较优化的绘图程序。

标题画面的图形代码排列见图6－10。

为使朋友们能够直接使用《魂斗罗》的绘图方法设计背景图案，特改造程序 No. 6—17为 No6—18。(仍以绘制《魂斗罗》的标题画面为例，使用原数据区。)

No. 6－18 背景绘图子程序(四)

```
7F00 206C7F     JSR $7F6C   ;保护现场，置数据区首址于
7F03 A900       LDA #$00    $50、$51
7F05 8D0120     STA $2001   ;关显示
7F08 A001       LDY #$01
7F0A B150       LDA ($50,Y)
7F0C 8D0620     STA $2006   ;置 PPU 地址高位
7F0F 88         DEY
7F10 B150       LDA ($50,Y)
```

```
7F12 8D0620      STA $2006   ;置 PPU 地址低位
7F15 A902        LDA #$02
7F17 20647F      JSR $7F64   ;调整取数地址
```

列次	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F
2000																															
2020																															
2040																															
2060																															
2080																															
20A0																															
20C0					02	03	04	05	06	07																					
20E0				11	12	13	14	15	16	17	18	19																			
2100			20	21	22	23	24	25	26	27	28	29																			
2120			60	61	62					08	09																				
2140			70	71	72			63	64	65	66	67	68	69	6A	6B	6C	6D	6E	6F	0A	0B	0C	0D							
2160			80	81	82			73	74	75	76	77	78	79	7A	7B	7C	7D	7E	7F	1A	1B	1C	1D	1E						
2180			90	91	92			83	84	85	86	87	88	89	8A		8C	8D	8E	8F	2A	2B	2C	2D	2E	2F					
21A0			A0	A1	A2	A3		93	94	95	96	97	98	99	9A		9C	9D	9E	9F	3A	3B	3C	3D	3E	3F					
21C0				B1	B2	B3	B4	B5	B6	B7	B8	B9	BA	BB																	
21E0						A4	A5	A6	A7	A8	A9																				
2200																			C4		D0	D1	D2								
2220																		D3	D4	D5	E0	E1	E2	E3							
2240				50	4C	4A	59		53	45	4C	45	43	54					E4	E5	F0	F1	F2								
2260																		F3	F4	F5	C5	C6	C7	C8	C9					
2280						31		50	4C	4A	59	45	52				CA	CB	CC	CD	CE					EF	D6				
22A0																D9	DA	DB	DC	DD	DE						E6				
22C0						32		50	4C	41	59	45	52	53		E9	EA	EB	EC	ED	EE						F5				
22E0																		D7	D8	E7	E8										
2300																															
2320																															
2340																															
2360																															
2380																															
23A0																															

注——空格部分为“00”

图6—10 《魂斗罗》标题画面

```
7F1A A000        LDY #$00
7F1C B150        LDA ($50,Y)
7F1E C9FF        CMP #$FF
7F20 D004        BNE $7F26   ;A≠FFH 转移取数送 PPU
7F22 20727F      JSR $7F72   ;A=FFH 结束送数,恢复现
                             场,开显示
```

```
7F25 60       RTS
7F26 C97F     CMP # $7F
7F28 F032     BEQ $7F5C  ;A=7FH 转移改变PPU 地址
7F2A A8       TAY
7F2B 101D     BPL $7F4A  ; A<80H 转移一个数连送
                         N 次
7F2D 297F     AND # $7F  ;A>7FH 则以其 D6~D0位
                         为计数送入 $52
7F2F 8552     STA $52
7F31 A001     LDY # $01
7F33 B150     LDA ($50,Y);取 N 个数送入 PPU
7F35 8D0720   STA $2007
7F38 C452     CPY $52
7F3A F003     BEQ $7F3F  ;达到数量调整取数地址
7F3C C8       INY
7F3D D0F4     BNE $7F33  ;未达到继续取数送数
7F3F A901     LDA # $01
7F41 18       CLC
7F42 6552     ADC $52
7F44 20647F   JSR $7F64  ;调整取数地址
7F47 4C1A7F   JMP $7F1A
7F4A A001     LDY # $01  ;一个数连送 N 次的处理
7F4C 8552     STA $52
7F4E B150     LDA ($50,Y)
7F50 A452     LDY $52
7F52 8D0720   STA $2007
7F55 88       DEY
```

```
7F56 D0FA       BNE $7F52
7F58 A902       LDA #$02
7F5A D0E8       BNE $7F44       ;送完后调整取数地址
7F5C A901       LDA #$01
7F5E 20647F     JSR $7F64       ;调整取数地址
7F61 4C087F     JMP $7F08       ;调整 PPU 地址
7F64 18         CLC             ;调整取数地址子程序
7F65 6550       ADC $50
7F67 9002       BCC $7F6B
7F69 E651       INC $51
7F6B 60         RTS
7F6C 48         PHA             ;保护现场子程序
7F6D 8A         TXA
7F6E 48         PHA
7F6F 98         TYA
7F70 48         PHA
7F71 A997       LDA #$97
7F73 8550       STA $50
7F75 A990       LDA #$90
7F77 8551       STA $51
7F79 60         RTS
7F7A A531       LDA $31         ;恢复现场子程序
7F7C 8D0120     STA $2001
7F7F 68         PLA
7F80 A8         TAY
7F81 68         PLA
7F82 AA         TAX
```

```
7F83 68          PLA
7F84 60          RTS
```

该子程序不必放入中断即可运行，读者只要把原数据区放入 RAM 的适当位置，运行 No. 6－18就可以绘出《魂斗罗》标题画面的图形结构(由于字模不同，绘出的画面是不一样的)。

第六节　间接分段绘图法

《魂斗罗》中的游戏画面都是采取间接分段绘图法绘制的。这种绘图方法的特点是：整幅画面分散绘制，一般一次中断或隔几次中断绘制一列或一行，横向卷动的画面每次绘一列，纵向滚动的画面每次绘一行；以七页为间接绘图数据缓存区，由当前中断把绘图数据送入七页，在下一次中断再把存于七页的数据送入 PPU。这种绘图方法可保证多幅画面连续绘制，产生连绵不绝的效果。如第一关开始时的游戏画面尤如幕布从左向右拉开，而第二关又如幕布由下向上提起。随着游戏的进程，画面一幕幕的位移，使人丝毫感觉不到画面的间断，整关游戏画面好象是一幅长卷图画，左卷右放，尽头方休。下面仅列出源绘图程序并加以简要说明，以供学习、参考。源程序见 No. 6－19。

程序 No. 6－19共分两段：第一段为 $CB60～$CBC5，完成把 PPU 地址、绘图数据从数据区读出，按照一定的格式送入七页；第二段为 $CC3F～$CC7E，完成把七页的绘图数据送入 PPU，绘制出游戏画面。

No. 6－19《魂斗罗》游戏画面绘制源程序

从数据区取数送入七页子程序

```
CB60 48        PHA            ;保存调该程序时的 A 值，
                              该值决定绘图的种类
CB61 A902      LDA # $02
CB63 8503      STA $03
CB65 A901      LDA # $01
CB67 20BDCB    JSR $CBBD      ;令 $700=1、$21=1
CB6A 68        PLA            ;弹出绘图类型数据
CB6B 8502      STA $02
CB6D 0A        ASL
CB6E AA        TAX            ;以绘图类型数据为地址偏移
                              量地址偏移量
CB6F BD62B2    LDA $B262,X
CB72 8500      STA $00
CB74 BD63B2    LDA $B263,X
CB77 8501      STA $01        ; 读出绘图数据区首址存入
                              $00、$01单元
CB79 A621      LDX $21
CB7B A000      LDY # $00
CB7D B100      LDA ($00,Y);读第一个数据
CB7F C8        INY
CB80 C9FF      CMP # $FF
CB82 F03F      BEQ $CBC3      ;若 A=FFH 结束读数
CB84 C9FE      CMP # $FE
CB86 F019      BEQ $CBA1      ; 若 A=FEH 令 $700+X=
                              FF、X+1→$21，结束取数
CB88 C9FD      CMP # $FD
CB8A F019      BEQ $CBA5      ; 若 A=FDH 令 $700+X=
```

```
                                    FF,进行第二次送数把数据
                                    送入七页
CB8C 9D0007  STA $0700,X
CB8F A502    LDA $02
CB91 100B    BPL $CB9E
CB93 A503    LDA $03        ;七页的数据格式为:
CB95 D005    BNE $CB9C      ;$700表示送数方式;
CB97 9D0007  STA $0700,X    ;$701表示一组送入 PPU
CB9A F002    BEQ $CB9E      ;的数据个数
CB9C C603    DEC $03        ;$702表示送入 PPU 数据的
                            组数
CB9E E8      INX            ;$703表示 PPU 地址高位
CB9F D0DC    BNE $CB7D      ;$704表示 PPU 地址低位
CBA1 A9FF    LDA #$FF       ;$705……为绘图数据
CBA3 D01A    BNE $CBBF      ;当一次送多组数据时,每一组
CBA5 A9FF    LDA #$FF       ;数据传送完毕后的下两个
CBA7 20BFCB  JSR $CBBF      ;数据,即为下一组绘图数据的
CBAA A902    LDA #$02       ;PPU 地址。
CBAC 8503    STA $03
CBAE A901    LDA #$01
CBB0 20BFCB  JSR $CBBF
CBB3 D0C8    BNE $CB7D
CBB5 A9FF    LDA #$FF
CBB7 D004    BNE $CBBD
CBB9 A900    LDA #$00
CBBB F000    BEQ $CBBD
CBBD A621    LDX $21
```

```
CBBF 9D0007  STA  $0700,X
CBC2 E8      INX
CBC3 8621    STX  $21
CBC5 60      RTS
```

取七页数据送入 PPU 子程序

```
CC3F A200    LDX  #$00
CC41 A5FF    LDA  $FF
CC43 2918    AND  #$18
CC45 8502    STA  $02
CC47 BC0007  LDY  $0700,X
CC4A F0D0    BEQ  $CC1C   ;$700=0不送数
CC4C A502    LDA  $02
CC4E 19C5CB  ORA  $CBC5,Y
CC51 8D0020  STA  $2000   ;设置送数方式
CC54 E8      INX
CC55 BD0007  LDA  $0700,X
CC58 8500    STA  $00     ;以$00为一组送数的个数计
                          数器
CC5A E8      INX
CC5B BD0007  LDA  $0700,X
CC5E 8501    STA  $01     ;以$01为本次送数的组数计
                          数器
CC60 A400    LDY  $00
CC62 E8      INX
CC63 BD0007  LDA  $0700,X
CC66 8D0620  STA  $2006   ;置 PPU 地址高位
CC69 E8      INX
```

```
CC6A BD0007 LDA  $0700,X
CC6D 8D0620 STA  $2006    ;置 PPU 地址低位
CC70 E8     INX
CC71 BD0007 LDA  $0700,X
CC74 8D0720 STA  $2007    ;取数送入 PPU
CC77 88     DEY
CC78 D0F6   BNE  $CC70    ;若本组未送完转$CC70继续
CC7A C601   DEC  $01      ;若本组已送完令组数计数器减一
CC7C D0E2   BNE  $CC60    ;若还有数据组转$CC60继续送数
CC7E 60     RTS           ;若数据组计数器为零结束送数
```

程序 No. 6－19通俗易懂，不多赘述。适用于电脑学习机的实用程序编写，留给读者练习。

第七章　卡通的控制

任天堂游戏中的角色统称为卡通(cartoon)。玩任天堂游戏的过程就是控制卡通动作的过程。

众所周知，卡通在整个游戏的过程中是最为活跃的部分，因而控制卡通的程序结构也就最复杂、程序量也最大。一般情况下，游戏软件中用于控制卡通动作的程序量约占总程序量的百分之五十左右(不包括数据区)，游戏越精彩，所占比例越大。如大家最宠爱的枪战游戏《魂斗罗》，软件容量为128K，其中控制程序约占 32K 字节，而用于卡通控制的程序则占到 20K 之多，比例高达百分之六十以上。由此可见，卡通动作的控制是十分复杂的。因此，限于篇幅，我们不可能对卡通的动作控制程序作全面的介绍。只能教给大家最基本的方法，由朋友们在编程实践中举一反三，领会掌握。

本章将首先指出 F BASIC 系统中卡通与任天堂游戏中卡通的区别；然后重点讨论操纵器的采样程序设计和卡通的定义方法；最后列出《大赛车》游戏中定义卡通的部分程序清单供欣赏参考。

第一节　两个系统中卡通的区别

通过以上讨论，我们已经熟悉了 F BASIC 工作系统中的卡通，并能得心应手的编写程序控制卡通的运动和显示，以实现不同的功能要求。同时，我们还都有深切的体会：用 F

BASIC 语言编写的卡通控制程序，运行起来速度特别慢，比任天堂游戏中的卡通要逊色得多，而且不能定义高点阵的卡通，无论怎样绞尽脑汁修改程序也难尽人意。因而，迫切地想了解任天堂游戏中对卡通的控制方法，梦寐以求地想把这种控制方法应用到 F BASIC 的工作系统中来。但是，任天堂游戏中的控制方法又独具特点，只有全面掌握两个系统对卡通控制方法的相似与区别，才能够应用灵活，行之有效。现将 F BASIC、任天堂游戏两个工作系统中卡通的特点归纳如下：

一、显示、动作卡通的分类

在 F BASIC 中，卡通被分为显示、动作两类，分别用 DEF SPRITE、DEF MOVE 指令来定义这两类卡通的显示和运动。而在任天堂游戏中，卡通是没有显示、动作之分的，它里面的卡通具有相同的共性和个性。F BASIC 把它分为两类的目的是为了便于指令控制，以实现不同的功能要求。这应该说是 F BASIC 对游戏机功能的扩展和创新。

二、定义卡通的数量

在任天堂游戏中，一般使用二页作为卡通的定义区。由于每一个卡通块（卡通显示的最小单位）要占用四个单元，故一次最多可定义 64 个卡通块。F BASIC 规定 16 点阵显示、动作卡通的数量各不允许超过 8 个；在任天堂游戏中则没有这种限制。只要一次定义的卡通块数量不超过 64 个，就可全部显示。

三、定义卡通的大小

在 F BASIC 中限定卡通最大为 16×16 点阵；在任天堂

游戏中则没有这种限制。只要实际需要，系统允许把64个卡通块定义成一个特大卡通。

四、卡通的运动控制

在F BASIC中，卡通一次定义后可一直使用，只要不改变它的运动状态，就不必重新定义；而在任天堂游戏中，卡通必须反复定义才能不断运动。也就是说，F BASIC中的卡通是由系统控制运动的；任天堂游戏中卡通的每一点运动都必须靠编程控制。因此，单从这一方面来说F BASIC中的卡通属智能型的；任天堂中的卡通属机械型的。但是，智能型的卡通由于程序已经固化，运动形式已经锁定，不能人工干预改变，因而它的原有优势反而变成了束缚它的桎梏，使它变成了机械型的。而任天堂中机械型的卡通，由于可通过编程控制其运动状态，因而，可纵横捭阖，灵活控制。从而扭转了它原来的机械劣势，而体现了领先的优势。

五、卡通的运动速度

在F BASIC中，运动卡通的最快速度为每次一点，即每中断一次运动一点，运动一点的最短时间为0·02秒。另外，卡通的运动方向只有系统设定好的8个方向(0°、45°、90°、135°、180°、225°、270°、315°)，用户不能再作其它选择。而任天堂游戏中，每中断一次可以对卡通重新定义一次，因而只要适当改变卡通的定义座标，就可以任意控制卡通的运动速度和运动方向。

六、卡通的姿势

F BASIC中16×16点阵卡通的组合是系统约定好的，

用户不能改变。如我们只能定义面向左或面向右、头朝上或头朝下的玛丽，而不能定义睡倒在地脸朝上或朝下的玛丽。同样也不能定义摔倒的丽莎或企鹅。虽然我们可以使用 DEF SPRITE 语句来选择组合，但组合的卡通又不能自动运动，因此其使用范围受到了极大的限制。造成这些束缚的原因是：卡通的字模数据已经设定，不能随意改变，用户只能使用系统固有的 256 个卡通字模（卡通块）组合卡通。因此，F BASIC 中的卡通是呆板的。在任天堂中，卡通字模可随时改变，如《魂斗罗》几乎每一关的开始、关底都要更换字模，其字模数据区约占 45K 字节。而《魂斗罗》二代、《脱狱》二代、《双截龙》等 256K 的游戏，其字模区竟高达 128K 字节。就因为这些游戏有如此丰富的字模，所以才能产生那么奇妙的背景画面和形神并貌的奇特卡通。

第二节　操纵器采样

在任天堂游戏的过程中，操纵器就是司令部，游戏者的意志就是通过操纵器传送给游戏机的。游戏者游戏水平的高低也全部从控制操纵器的过程中体现出来。可见，操纵器的采样处理直接关系到游戏者意志的完整转达。因此，操纵器采样处理程序性能的好坏直接体现了一个游戏软件的设计水平，也直接影响着游戏软件的功能评价。

操纵器的采样是通过读系统软开关（寄存器）$4016、$4017 实现的。针对游戏控制过程的特点，一般要求能够读出操纵器的瞬间值，以满足快速变化控制的需要；能够读出操纵器的稳键值，以满足选项过程中控制选项的需要；同时还要求能够输出操纵器的单步值和连续值，以实现不同的游

戏控制功能。

每个软件的操纵器采样程序模式虽然比较相近，但也各有千秋。下面仅介绍《大赛车》、《小学算术一年级》、《魂斗罗》游戏中三个较典型的操纵器采样程序，最后给出 BS·2A F BASIC 系统操纵器采样子程序的调用方法。

一、《大赛车》的操纵器采样

《大赛车》为单人游戏，游戏中只对 0# 操纵器采样。程序如 No. 7－1。

No. 7－1《大赛车》操纵器采样程序

```
C066 48       PHA
C067 8A       TXA
C068 48       PHA
C069 98       TYA
C06A 48       PHA
C06B A900     LDA  # $00
C06D 8D0320   STA  $2003
C070 A54F     LDA  $4F
C072 2901     AND  # $01
C074 0902     ORA  # $02
C076 8D1440   STA  $4014
C079 A001     LDY  # $01
C07B 8C1640   STY  $4016
C07E 88       DEY
C07F 8C1640   STY  $4016
C082 AD1640   LDA  $4016
C085 2903     AND  # $03
```

```
C087 856D     STA  $6D      ;A 键代码存 $6D
C089 AD1640   LDA  $4016
C08C 2903     AND  #$03
C08E 856E     STA  $6E      ;B 键代码存 $6E
C090 AD1640   LDA  $4016
C093 2903     AND  #$03
C095 856B     STA  $6B      ;SELECT 键代码存 $6B
C097 AD1640   LDA  $4016
C09A 2903     AND  #$03
C09C 856C     STA  $6C      ;START 键代码存 $6C
C09E A001     LDY  #$01
C0A0 AD1640   LDA  $4016
C0A3 2903     AND  #$03
C0A5 D003     BNE  $C0AA
C0A7 88       DEY
C0A8 10F6     BPL  $C0A0
C0AA 846F     STY  $6F      ;上、下键代码存 $6F
C0AC A001     LDY  #$01
C0AE AD1640   LDA  $4016
C0B1 2903     AND  #$03
C0B3 D003     BNE  $C0B8
C0B5 88       DEY
C0B6 10F6     BPL  $C0AE
C0B8 8470     STY  $70      ;左、右键代码存 $70
C0BA A471     LDY  $71
C0BC F00A     BEQ  $C0C8
C0BE 846D     STY  $6D
```

```
C0C0 88      DEY
C0C1 846E    STY $6E
C0C3 88      DEY
C0C4 8470    STY $70
C0C6 846F    STY $6F
C0C8 A901    LDA #$01
C0CA 8516    STA $16
C0CC 68      PLA
C0CD A8      TAY
C0CE 68      PLA
C0CF AA      TAX
C0D0 68      PLA
C0D1 40      RTI
```

《大赛车》的操纵器采样是在中断处理中进行的，读出的各按键值存于零页单元$6B～$70中，其对应关系见表7—1。

表7—1 《大赛车》操纵器键码对照表

按键名称	$6B	$6C	$6D	$6E	$6F	$70
上	00	00	00	00	01	FF
下	00	00	00	00	00	FF
左	00	00	00	00	FF	01
右	00	00	00	00	FF	00
A	00	00	01	00	FF	FF
B	00	00	00	01	FF	FF
SELECT	01	00	00	00	FF	FF
START	00	01	00	00	FF	FF

程序No.7—1很好理解，不多赘述。

二、《小学算术一年级》的操纵器采样

《小学算术一年级》的操纵器采样程序为一独立的子程序，它能一次读出两个操纵器的按键代码。程序见 No. 7—2。

No. 7—2《小学算术一年级》操纵器采样程序

```
AEEE A201       LDX  # $01
AEF0 8E1640     STX  $4016
AEF3 CA         DEX
AEF4 8E1640     STX  $4016
AEF7 20FBAE     JSR  $AEFB      ;读 0# 操纵器
AEFA E8         INX             ;读 1# 操纵器
AEFB A008       LDY  # $08
AEFD 48         PHA
AEFE BD1640     LDA  $4016,X
AF01 851E       STA  $1E
AF03 4A         LSR
AF04 051E       ORA  $1E
AF06 4A         LSR
AF07 68         PLA
AF08 2A         ROL
AF09 88         DEY
AF0A D0F1       BNE  $AEFD
AF0C 851E       STA  $1E
AF0E B506       LDA  $06,X
AF10 49FF       EOR  # $FF
AF12 251E       AND  $1E
AF14 9508       STA  $08,X
```

```
AF16 A51E       LDA  $1E
AF18 9506       STA  $06,X
AF1A 60         RTS
```

程序 No. 7－2 读出的按键代码存入 $06～$09 单元中，其中 0＃操纵器代码存 $06、$08 单元；1＃操纵器代码存 $07、$09 单元。为什么一个操纵器代码存入两个单元中呢？分析一下 No. 7－2 就可发现(以 0＃操纵器为例)，$08 单元存按键的瞬间值，即存按键脉冲上升沿，且只存一个中断周期，一个中断周期过去后其值又为零；而 $06 单元虽然也存按键脉冲的上升沿，但可长期保存，只要不改按其它键则按键代码一直保持，直到改按新的按键为止。按键代码的对应关系见表 7－2。

表 7－2 《小学算术一年级》操纵器按键代码表

按键	不按	上	下	左	右	A	B	SEL－ECT	STA－RT
代码	00	08	04	02	01	80	40	20	10

三、《魂斗罗》的操纵器采样

《魂斗罗》游戏的操纵器采样处理也是一个独立的子程序，它被安排在中断处理中调用。采样中以 $1D 单元为读 0＃、1＃操纵器的转换开关，可一次读出两个操纵器的按键代码。程序见 No. 7－3。

No. 7－3《魂斗罗》操纵器采样程序

```
C35B 2097C3     JSR  $C397    ;操纵器采样子程序
C35E A504       LDA  $04
C360 8500       STA  $00
```

```
C362 A505     LDA $05
C364 8501     STA $01
C366 2097C3   JSR $C397   ;第二次采样认定
C369 A201     LDX #$01
C36B B504     LDA $04,X
C36D D500     CMP $00,X
C36F F004     BEQ $C375
C371 B5F9     LDA $F9,X
C373 9504     STA $04,X
C375 CA       DEX
C376 10F3     BPL $C36B
C378 A51D     LDA $1D     ;由$1D控制转换
C37A 2904     AND #$04
C37C D006     BNE $C384
C37E A504     LDA $04
C380 0505     ORA $05
C382 8504     STA $04
C384 A201     LDX #$01
C386 B504     LDA $04,X
C388 A8       TAY
C389 55F9     EOR $F9,X
C38B 3504     AND $04,X
C38D 95F5     STA $F5,X
C38F 94F1     STY $F1,X
C391 94F9     STY $F9,X
C393 CA       DEX
C394 10F0     BPL $C386
```

```
C396 60         RTS
C397 A201       LDX #$01   ;采样子程序
C399 8E1640     STX $4016
C39C CA         DEX
C39D 8E1640     STX $4016
C3A0 A008       LDY #$08
C3A2 AD1640     LDA $4016
C3A5 8507       STA $07
C3A7 4A         LSR
C3A8 0507       ORA $07
C3AA 4A         LSR
C3AB 2604       ROL $04
C3AD AD1740     LDA $4017
C3B0 8507       STA $07
C3B2 4A         LSR
C3B3 0507       ORA $07
C3B5 4A         LSR
C3B6 2605       ROL $05
C3B8 88         DEY
C3B9 D0E7       BNE $C3A2
C3BB 60         RTS
```

程序 No. 7－3 读出的按键代码与表 7－2 相同，读键过程留给读者分析。

四、F BASIC 操纵器采样子程序的调用

在 F BASIC 指令中有两条操纵器采样指令：

STICK

STRIG

这两条指令分别完成操纵器的方向键和功能键的采样。STICK 指令的返回值与上、下、左、右键的对应关系是 8、4、2、1；STRIG 指令的返回值与 A、B、SELECT、START 键的对应关系也是 80、40、20、10。我们在进行机器语言编程时可以调用这两条指令的读键子程序。为便于其它系统的调用，特将该子程序列于 No. 7－4。

No. 7－4 操纵器采样子程序

```
A682 A917     LDA #$17      ;读 1# 操纵器的入口
A684 AEA916   LDX $16A9     ;读 0# 操纵器的入口
A687 8519     STA $19
A689 A940     LDA #$40
A68B 851A     STA $1A
A68D A905     LDA #$05
A68F 8D1640   STA $4016
A692 EA       NOP
A693 EA       NOP
A694 A904     LDA #$04
A696 8D1640   STA $4016
A699 EA       NOP
A69A EA       NOP
A69B A208     LDX #$08
A69D A000     LDY #$00
A69F B119     LDA ($19,Y)
A6A1 4A       LSR
A6A2 2690     ROL $90
A6A4 EA       NOP
```

```
A6A5 EA      NOP
A6A6 CA      DEX
A6A7 D0F6    BNE $A69F
A6A9 60      RTS
```

No. 7—4 的操纵器采样子程序可同时读出方向键、功能键的键值，其返回值存于 $90 单元，按键与键值的对应关系同表 7—2。

程序 No. 7—4 的调用方法如下：

当对 0# 操纵器采样时要调 $A685；

当对 1# 操纵器采样时要调 $A682；

调用前要注意保护好 $19、$1A、$90 单元的现场。

第三节　定义卡通

卡通的定义过程就是把卡通块的四个参数(Y 座标、卡通图块代码、卡通的显示状态、X 座标)写入选定的卡通定义区。综合比较各种软件的卡通定义程序，唯有 F BASIC 的对话工作系统中的卡通定义程序独立性最强，最利于理解，最方便移植。因此，我们仅以它为例介绍卡通定义程序的编写。

对话工作系统的卡通定义过程共分两步完成：首先把卡通的显示状态送入定义区，并置 Y 座标为零；然后再把卡通的座标参数和图形代码依次送入卡通定义区。下面以例 7—1 为例讨论这一定义过程：

例 7—1 用八个卡通块定义一个 16×32 点阵的卡通，即横向为两个卡通块、纵向为四个卡通块(《魂斗罗》游戏中的主人翁即为 16×32 点阵的卡通)。

一、卡通显示状态的定义

1. 卡通显示状态的定义程序

卡通显示状态的定义程序见 No. 7－5。

No. 7－5 卡通的显示状态定义子程序

```
EC86 A000     LDY #$00
EC88 B100     LDA ($00,Y)
EC8A C9FF     CMP #$FF
EC8C F013     BEQ $ECA1      ;读数为 FFH 时结束定义
EC8E 8502     STA $02        ;状态参数存入 $02
EC90 C8       INY
EC91 B100     LDA ($00,Y)
EC93 8503     STA $03        ;定义卡通块的数量存入
                               $03
EC95 C8       INY
EC96 B100     LDA ($00,Y)
EC98 8504     STA $04
EC9A 20A2EC   JSR $ECA2      ;调状态定义子程序
EC9D C8       INY
EC9E 4C88EC   JMP $EC88
ECA1 60       RTS
```

定义卡通状态子程序

```
ECA2 98       TYA
ECA3 48       PHA
ECA4 A902     LDA #$02
ECA6 8505     STA $05
ECA8 A603     LDX $03
```

```
ECAA A000    LDY  # $00
ECAC A900    LDA  # $00
ECAE 9104    STA  ($04,Y)      ;对 Y 座标单元置零
ECB0 C8      INY
ECB1 C8      INY
ECB2 A502    LDA  $02
ECB4 9104    STA  ($04,Y)      ;状态参数送入定义区
ECB6 C8      INY
ECB7 C8      INY
ECB8 CA      DEX
ECB9 D0F1    BNE  $ECAC
ECBB 68      PLA
ECBC A8      TAY
ECBD 60      RTS
```

2. 定义卡通显示状态子程序的调用要求

在调用 No. 7－5 之前要先对如下单元置入初值：

(1) $00、$01 两单元置入数据区首址。

(2)对数据区置入状态参数。数据区的数据结构如下：每三个数据为一组，第一个数据为卡通状态；第二个数据为本次定义卡通块的数量；第三个数据为卡通块在卡通定义区的地址低位；最后还要置入定义结束标志 FFH。

对于例 7－1，假如我们选定卡通的状态参数为 03；其定义参数放在 $210～$22F 连续的 20H 个单元中，并选定数据区首址为 $7E00。那么，数据区结构如下：

7E00 03 08 10 FF

对 $00 单元置入 00、对 $01 单元置入 7EH，使其指向 $7E00。

3. 定义卡通显示状态的操作过程

现在我们调用程序 No. 7－5，分析卡通状态的定义过程：

(1)执行 $EC86，从数据区读出卡通的状态参数存入 $02 单元，有 $02＝03；读出定义卡通块的数量存入 $03 单元，有 $03＝08；读出卡通定义区地址低位存入 $04 单元，有 $04＝10。

(2)调 $ECA2 子程序，令 $05＝02，使 $04、$05 指向该卡通定义区的首址 $210。

(3)执行 $ECAC，对 Y 座标单元置入零。

(4)执行 $ECB2，把卡通的状态参数存入二页。

(5)令 X 减一，X 不等于零则返回 $ECAC，继续定义下一个卡通块的状态。

(6)若 X 等于零，则结束状态定义，返回调用程序 $EC9A。

(7)令 Y 加一，转 $EC88，继续从数据区读数，若读出的数据不等于 FFH，则继续定义下一个卡通的状态，现在读出的数据是 FFH，则结束定义。

通过调用程序 No. 7－5，有关二页单元的内容如下：

$210＝00 $212＝03 $214＝00 $216＝03

$218＝00 $21A＝03 $21C＝00 $21E＝03

$220＝00 $222＝03 $224＝00 $226＝03

$228＝00 $22A＝03 $22C＝00 $22E＝03

二、卡通座标、代码的定义

1. 定义卡通座标、代码的子程序

定义卡通座标、代码的子程序见 No. 7－6。

No. 7－6 定义卡通座标、代码子程序

```
ECBF 48      PHA
ECC0 98      TYA
ECC1 48      PHA
ECC2 A902    LDA #$02
ECC4 8505    STA $05
ECC6 A90F    LDA #$0F
ECC8 2503    AND $03
ECCA 8507    STA $07      ;取$03单元的低位置入$07
ECCC A503    LDA $03
ECCE 4A      LSR
ECCF 4A      LSR
ECD0 4A      LSR
ECD1 4A      LSR
ECD2 8506    STA $06      ;取$03单元的高位置入$06
ECD4 AA      TAX
ECD5 A900    LDA #$00
ECD7 18      CLC
ECD8 6507    ADC $07
ECDA CA      DEX
ECDB D0FB    BNE $ECD8
ECDD 8508    STA $08
ECDF 20EAEC  JSR $ECEA    ;定义卡通的代码
ECE2 2005ED  JSR $ED05    ;定义卡通的座标
ECE5 68      PLA
ECE6 A8      TAY
ECE7 68      PLA
ECE8 AA      TAX
```

```
ECE9 60      RTS
     定义卡通的代码子程序
ECEA A502    LDA $02
ECEC A608    LDX $08
ECEE A001    LDY #$01
ECF0 9104    STA ($04,Y)   ;卡通的代码存入定义区
ECF2 18      CLC
ECF3 6901    ADC #$01
ECF5 C8      INY
ECF6 48      PHA
ECF7 B104    LDA ($04,Y)
ECF9 293F    AND #$3F
ECFB 9104    STA ($04,Y)   ;限定卡通的状态参数不大
                           于 3FH
ECFD 68      PLA
ECFE C8      INY
ECFF C8      INY
ED00 C8      INY
ED01 CA      DEX
ED02 D0EC    BNE $ECF0
ED04 60      RTS
     定义卡通的座标子程序
ED05 A000    LDY #$00
ED07 A606    LDX $06
ED09 A500    LDA $00
ED0B 8509    STA $09
ED0D A501    LDA $01
```

```
ED0F 9104    STA ($04,Y)   ;定义卡通的Y座标
ED11 C8      INY
ED12 C8      INY
ED13 C8      INY
ED14 A509    LDA $09
ED16 9104    STA ($04,Y)   ;定义卡通的X座标
ED18 18      CLC
ED19 6908    ADC #$08
ED1B 8509    STA $09
ED1D C8      INY
ED1E CA      DEX
ED1F D0EC    BNE $ED0D
ED21 A501    LDA $01
ED23 18      CLC
ED24 6908    ADC #$08
ED26 8501    STA $01
ED28 C607    DEC $07
ED2A D0DB    BNE $ED07
ED2C 60      RTS
```

2. 定义卡通座标、代码子程序的调用要求

调用程序 No. 7－6 之前，必须先对如下单元置入数据：

(1)对 $00 单元置入卡通的 X 座标值；对 $01 单元置入卡通的 Y 座标值。

(2)对 $02 单元置入第一个卡通块的代码(以后卡通块的代码则顺序置入)。

(3)对 $03 单元置入卡通的定义指南，所谓卡通的定义指南的结构为：其高位标志着横排卡通块的数量，其低位标

志着竖排卡通块的数量。

(4) 对 $04 单元置入卡通块定义区的地址低位。

(5)对 $05 单元置入卡通块定义区的地址高位。

假定例 7－1 中卡通的 X 座标为 60H、Y 座标为 90H、第一个卡通块的代码为 BCH，那么 $00～ $05 单元的数据如下：

$00＝60 $01＝90 $02＝BC

$03＝24 $04＝10 $05＝02

3. 定义卡通座标、代码的操作过程

现在分析定义卡通座标、代码的过程：

(1)执行 $ECBE，把 $03 单元的高位 02 置入 $06 单元、低位 04 置入 $ $07 单元；

由 $06 控制 $07 的累加次数，将累加之和 08 置入 $08 单元，以控制定义卡通块的总数量。

(2)调 $ECEA，把卡通代码置入定义区，同时限定卡通的状态参数不大于 3FH。

(3)调 $ED05，把卡通的 Y 座标、X 座标参数置入定义区。

通过程序 No. 7－5、No. 7－6 的处理，卡通定义区的有关单元置入了如下参数：

$210＝90	$214＝90
$211＝BC	$215＝BD
$212＝03	$216＝03
$213＝60	$217＝68
* * *	* *
$218＝98	$21C＝98
$219＝BE	$21D＝BF
$21A＝03	$21E＝03

$21B=60	$21F=68
* * *	* *
$220=A0	$224=A0
$221=C0	$225=C1
$222=03	$226=03
$223=60	$227=68
* * *	* *
$228=A8	$22C=A8
$229=C2	$22D=C3
$22A=03	$22E=03
$22B=60	$22F=68

以上定义了在螃蟹背上站立一个乌龟的16×32点阵的大卡通，由于卡通字库的限制，这仅仅是示意性的。

在实际应用中，我们必须周密的设计数据区，并配置一个根据按键的种类分别对$00～$05单元赋值的控制程序，以保证卡通按照操纵器的指令动作。

三、实用的卡通定义子程序

享有在BS·2A版本的F BASIC工作系统的朋友，可以直接调用子程序$EC86、$ECBE，从而可极大的简化定义程序。没有BS·2A版本的朋友则只能根据No.7－5、No.7－6另编两段子程序供调用。实用的卡通定义子程序见No.7－7。

1. 工作单元的选取

在编写定义卡通的子程序之前，要先选好一部分零页单元作为卡通参数存储器，选取原则是避开系统冲突，优先选用闲置的零页单元。我们选取$50、$51、$77～$7B共7

个零页单元为定义工作单元，它们的分工如下：

$50 作为状态定义开关。一般情况下，卡通的状态一经定义后，一般不会经常改变，所以设置一个软开关以减少子程序的调用开销。

$51 单元用于存储卡通状态定义数据区地址的偏移量，以便从数据区查出定义卡通状态的数据区首址存入$00、$01 单元。

$77 作为卡通 X 座标的存储器。

$78 作为卡通 Y 座标的存储器。

$79 作为卡通块的图形库代码存储器。

$7A 用于存储一次定义卡通块的数量和排列形式的数据。

$7B 用于存储定义卡通块的二页地址低位。

2. 实用卡通定义子程序

实用卡通定义子程序见 No. 7—7。

No. 7—7 实用卡通定义子程序

```
7F00 20357F     JSR $7F35   ;保护现场
7F03 A550       LDA $50
7F05 F00F       BEQ $7F16  ;检查卡通状态定义开关
7F07 A451       LDY $51
7F09 B9807F     LDA $7F80,Y
7F0C 8500       STA $00
7F0E B9817F     LDA $7F81,Y
7F11 8501       STA $01
7F13 2086EC     JSR $EC86   ;定义卡通状态
7F16 A577       LDA $77
7F18 8500       STA $00
```

```
7F1A A578      LDA  $78
7F1C 8501      STA  $01
7F1E A579      LDA  $79
7F20 8502      STA  $02
7F22 A57A      LDA  $7A
7F24 8503      STA  $03
7F26 A57B      LDA  $7B
7F28 8504      STA  $04
7F2A A902      LDA  #$02
7F2C 8505      STA  $05
7F2E 20BEEC    JSR  $ECBE
7F31 20437F    JSR  $7F43   ;恢复现场
7F34 60        RTS
```

保护现场子程序

```
7F35 48        PHA
7F36 8A        TXA
7F37 48        PHA
7F38 98        TYA
7F39 48        PHA
7F3A A209      LDX  #$09
7F3C B500      LDA  $00,X
7F3E 48        PHA
7F3F CA        DEX
7F40 10FA      BPL  $7F3C
7F42 60        RTS
```

恢复现场子程序

```
7F43 A200      LDX  #$00
```

```
7F45 68        PLA
7F46 9500      STA $00,X
7F48 E8        INX
7F49 E00A      CPX #$0A
7F4B 90F8      BCC $7F45
7F4D 68        PLA
7F4E A8        TAY
7F4F 68        PLA
7F50 AA        TAX
7F51 68        PLA
7F52 60        RTS
```

程序 No. 7—7 中选择了卡通状态数据区首址为 $7F80，用户可以自由选择。选定后只要更改 $7F09、$7F0E 两程序行的数据地址就可以了。

第四节《大赛车》卡通定义程序简介

《大赛车》游戏中的主卡通——赛车，一般由十六至十八个卡通块组成(依据赛车直行和转弯的状态)。其定义程序为 $D2B2～$D400、其数据区在 $DA13～$DD0B，现简单介绍如下。

一、赛车定义子程序清单

赛车定义子程序见 No. 7—8。

No. 7—8 赛车定义子程序

```
D2B2 A550      LDA $50
D2B4 F00A      BEQ $D2C0
```

```
D2B6 A551     LDA  $51
D2B8 C9A0     CMP  #$A0
D2BA 9004     BCC  $D2C0
D2BC A901     LDA  #$01
D2BE 8575     STA  $75
D2C0 A556     LDA  $56
D2C2 F003     BEQ  $D2C7
D2C4 4C54F2   JMP  $F254    ;撞车时转$F254定义爆炸
D2C7 A55A     LDA  $5A
D2C9 F003     BEQ  $D2CE
D2CB 4CB5F3   JMP  $F3B5    ;游戏成功时定义赛车越远
                            越小
D2CE A90B     LDA  #$0B
D2D0 A470     LDY  $70      ;根据$70单元的数值调整
                            $7C
D2D2 3016     BMI  $D2EA
D2D4 F00A     BEQ  $D2E0
D2D6 C67C     DEC  $7C
D2D8 A57C     LDA  $7C
D2DA 100E     BPL  $D2EA
D2DC A900     LDA  #$00
D2DE F00A     BEQ  $D2EA
D2E0 E67C     INC  $7C
D2E2 A57C     LDA  $7C
D2E4 C916     CMP  #$16
D2E6 9002     BCC  $D2EA
D2E8 A916     LDA  #$16
```

```
D2EA 857C    STA $7C
D2EC A550    LDA $50      ;根据$50、$51 调整$55、
                          $54
D2EE 4A      LSR
D2EF A551    LDA $51
D2F1 6A      ROR
D2F2 18      CLC
D2F3 6555    ADC $55
D2F5 8555    STA $55
D2F7 9006    BCC $D2FF
D2F9 A554    LDA $54
D2FB 4905    EOR #$05
D2FD 8554    STA $54
D2FF A47C    LDY $7C      ;  根据$7C 的取值调整
                          $19,然后以$19 为地
                          址偏移量
D301 B945DA  LDA $DA45,Y
D304 18      CLC
D305 6554    ADC $54      ;对卡通数据地址存储单元
                          $22、$23 赋值;
D307 8519    STA $19
D309 A8      TAY
D30A B913DA  LDA $DA13,Y
D30D 8522    STA $22      ;对同时送二页、三页定义卡
                          通块的数量存储单元$17
                          赋值
D30F B91DDA  LDA $DA1D,Y
```

```
D312 8523      STA  $23
D314 B927DA    LDA  $DA27,Y; 对只送二页卡通数量存储
                             单元$17 赋值
D317 8517      STA  $17
D319 B931DA    LDA  $DA31,Y; 对只送三页卡通数量存储
                             单元$18 赋值
D31C 8518      STA  $18
D31E B93BDA    LDA  $DA3B,Y
D321 851A      STA  $1A
D323 A208      LDX  #$08
D325 A55C      LDA  $5C
D327 F052      BEQ  $D37B
D329 A57B      LDA  $7B      ; $5C≠0 时定义刹车时轮
                             胎后的飞溅尘土
D32B 4901      EOR  #$01
D32D 857B      STA  $7B
D32F A9D2      LDA  #$D2
D331 2008D1    JSR  $D108
D334 A57B      LDA  $7B
D336 F002      BEQ  $D33A
D338 A940      LDA  #$40
D33A A20A      LDX  #$0A
D33C 2008D1    JSR  $D108
D33F A57B      LDA  $7B
D341 D007      BNE  $D34A
D343 A966      LDA  #$66
D345 A26E      LDX  #$6E
```

```
D347 4C4ED3   JMP  $D34E
D34A A992     LDA  #$92
D34C A28A     LDX  #$8A
D34E 8D0B02   STA  $020B
D351 8D0B03   STA  $030B
D354 8E0F02   STX  $020F
D357 8E0F03   STX  $030F
D35A A55D     LDA  $5D
D35C A8       TAY
D35D 0A       ASL
D35E 18       CLC
D35F 69F3     ADC  #$F3
D361 8D0902   STA  $0209
D364 8D0903   STA  $0309
D367 6901     ADC  #$01
D369 8D0D02   STA  $020D
D36C 8D0D03   STA  $030D
D36F A57B     LDA  $7B
D371 D006     BNE  $D379
D373 C8       INY
D374 98       TYA
D375 2903     AND  #$03
D377 855D     STA  $5D
D379 A210     LDX  #$10
D37B A000     LDY  #$00     ;定义赛车
D37D B122     LDA  ($22,Y)
D37F 18       CLC
```

```
D380 69B8      ADC # $B8
D382 9D0002    STA $0200,X
D385 9D0003    STA $0300,X
D388 E8        INX
D389 C8        INY
D38A B122      LDA ($22,Y)
D38C 9D0002    STA $0200,X
D38F 9D0003    STA $0300,X
D392 E8        INX
D393 C8        INY
D394 B122      LDA ($22,Y)
D396 9D0002    STA $0200,X
D399 9D0003    STA $0300,X
D39C E8        INX
D39D C8        INY
D39E B122      LDA ($22,Y)
D3A0 18        CLC
D3A1 6980      ADC # $80
D3A3 9D0002    STA $0200,X
D3A6 9D0003    STA $0300,X
D3A9 E8        INX
D3AA C8        INY
D3AB C417      CPY $17
D3AD 90CE      BCC $D37D
D3AF 862B      STX $2B
D3B1 B122      LDA ($22,Y) ;定义卡通数据只送二页的处理
D3B3 18        CLC
```

```
D3B4 69B8      ADC #$B8
D3B6 9D0002    STA $0200,X
D3B9 E8        INX
D3BA C8        INY
D3BB B122      LDA ($22,Y)
D3BD 9D0002    STA $0200,X7
D3C0 E8        INX
D3C1 C8        INY
D3C2 B122      LDA ($22,Y)
D3C4 9D0002    STA $0200,X
D3C7 E8        INX
D3C8 C8        INY
D3C9 B122      LDA ($22,Y)
D3CB 18        CLC
D3CC 6980      ADC #$80
D3CE 9D0002    STA $0200,X
D3D1 E8        INX
D3D2 C8        INY
D3D3 C41A      CPY $1A
D3D5 90DA      BCC $D3B1
D3D7 A62B      LDX $2B      ;定义卡通的数据只送三页
                              的处理
D3D9 B122      LDA ($22,Y)
D3DB 18        CLC
D3DC 69B8      ADC #$B8
D3DE 9D0003    STA $0300,X
D3E1 E8        INX
```

```
D3E2 C8       INY
D3E3 B122     LDA ($22,Y)
D3E5 9D0003   STA $0300,X
D3E8 E8       INX
D3E9 C8       INY
D3EA B122     LDA ($22,Y)
D3EC 9D0003   STA $0300,X
D3EF E8       INX
D3F0 C8       INY
D3F1 B122     LDA ($22,Y)
D3F3 18       CLC
D3F4 6980     ADC #$80
D3F6 9D0003   STA $0300,X
D3F9 E8       INX
D3FA C8       INY
D3FB C418     CPY $18
D3FD 90DA     BCC $D3D9
D3FF 862B     STX $2B
D341 60       RTS
```

二、定义赛车卡通的数据

以下由$DA13～$DA26组成十组数据地址用于送入$22、$23单元，随后的十个数据送入$17单元、十个数据送入$18单元、十个数据送入$1A单元。

```
DA13        5C A4 E8 28 6C B4 FC 40 80 C4 DADADA
DA20 DBDBDBDBDCDCDC 30 2C 30 2C 30 30 2C 30 2C
```

DA30 30 48 44 40 44 48 48 44 40 44 48 3C 38 38 38 3C
DA40 3C 38 38 38 3C 00 01 01 01 01 01 01 01 01 01 01
DA50 02 03 03 03 03 03 03 03 03 03 03 04

以下为定义赛车直行、左转、右转共十种形状的数据

DA5C 00 1F 40 E8
DA60 00 1D 40 F8 00 1C 40 00 08 23 40 F0 08 22 40 F8
DA70 08 21 40 00 10 28 40 F8 10 27 40 00 10 26 40 08
DA80 18 2D 40 F8 18 2C 40 00 18 2B 40 08 08 20 40 08
DA90 10 25 40 10 18 2A 40 10 08 24 40 E8 10 29 40 F0
DAA0 18 2E 40 F0 00 00 40 EE 00 0A 40 06 08 11 40 F6
DAB0 08 10 40 FE 08 0F 40 06 10 16 40 F6 10 15 40 FE
DAC0 10 14 40 06 18 1A 40 F6 18 19 40 FE 18 18 40 06
DAD0 08 0E 40 0E 10 13 40 0E 18 07 40 0E 08 12 40 EE
DAE0 10 17 40 EE 18 1B 40 EE 00 00 00 F0 00 00 40 08
DAF0 08 02 00 F0 08 03 00 F8 08 03 40 00 08 02 40 08
DB00 10 05 00 F4 10 06 00 FC 10 05 40 04 18 08 00 F4
DB10 18 09 00 FC 18 08 40 04 10 04 00 EC 18 07 00 EC
DB20 10 04 40 0C 18 07 40 0C 00 0A 00 F2 00 00 00 0A
DB30 08 0F 00 F2 08 10 00 FA 08 11 00 02 10 14 00 F2
DB40 10 15 00 FA 10 16 00 02 18 18 00 F2 18 19 00 FA
DB50 18 1A 00 02 08 0E 00 EA 10 13 00 EA 18 07 00 EA
DB60 08 12 00 0A 10 17 00 0A 18 1B 00 0A 00 1C 00 F8
DB70 00 1D 00 00 00 1F 00 10 08 21 00 F8 08 22 00 00
DB80 08 23 00 08 10 26 00 F0 10 27 00 F8 10 28 00 00
DB90 18 2B 00 F0 18 2C 00 F8 18 2D 00 00 08 20 00 F0
DBA0 10 25 00 E8 18 2A 00 E8 08 24 00 10 10 29 00 08
DBB0 18 2E 00 08 00 3B 40 E8 00 1D 40 F8 00 1E 40 00
DBC0 08 23 40 F0 08 22 40 F8 08 3A 40 00 10 3E 40 F8

```
DBD0  10 27 40 00 10 26 40 08 18 2D 40 F8 18 2C 40 00
DBE0  18 2B 40 08 08 20 40 08 10 3D 40 10 18 2A 40 10
DBF0  08 3C 40 E8 10 3F 40 F0 18 2E 40 F0 00 00 40 EE
DC00  00 0A 40 06 08 11 40 F6 08 10 40 FE 08 33 40 06
DC10  10 37 40 F6 10 15 40 FE 10 14 40 06 18 1A 40 F6
DC20  18 19 40 FE 18 18 40 06 08 0E 40 0E 10 36 40 0E
DC30  18 07 40 0E 08 35 40 EE 10 38 40 EE 18 1B 40 EE
DC40  00 00 00 F0 00 00 40 08 08 30 00 F0 08 03 00 F8
DC50  08 03 40 00 08 30 40 08 10 05 00 F4 10 06 00 FC
DC60  10 05 40 04 18 08 00 F4 18 09 00 FC 18 08 40 04
DC70  10 31 00 EC 18 07 00 EC 10 31 40 0C 18 07 40 0C
DC80  00 0A 00 F2 00 00 00 0A 08 33 00 F2 08 10 00 FA
DC90  08 11 00 02 10 14 00 F2 10 15 00 FA 10 37 00 02
DCA0  18 18 00 F2 18 19 00 FA 18 1A 00 02 08 0E 00 EA
DCB0  10 36 00 EA 18 07 00 EA 08 35 00 0A 10 38 00 0A
DCC0  18 1B 00 0A 00 1E 00 F8 00 1D 00 00 00 3B 00 10
DCD0  08 3A 00 F8 08 22 00 00 08 23 00 08 10 26 00 F0
DCE0  10 27 00 F8 10 3E 00 00 18 2B 00 F0 18 2C 00 F8
DCF0  18 2D 00 00 08 20 00 F0 10 3D 00 E8 18 2A 00 E8
DD00  08 3C 00 10 10 3F 00 08 18 2E 00 08
```

程序 No. 7－8 仅仅是赛车正常行驶状态的定义处理，并不包括其它干扰赛车、路标、指示标语、撞车爆炸、赛车越远越小等卡通的定义，也不包括撞车、下路等情况的判断处理。由此可见，在任天堂游戏中卡通的控制处理是比较复杂的。由于 No. 7－8 是从游戏软件中截取的一段，前后连贯性较差，故仅作简单的介绍，旨在使朋友们初步接触一下实实在在的任天堂游戏，对它有一个初步的认识。详细的讨论请参阅拙作《任天堂游戏编程探密》一书。

第八章　背景画面的分裂位移与扭曲

在大多数任天堂游戏节目中，都采用了整幅背景画面的位移技术(横向卷动或纵向滚动)，如著名枪战游戏《魂斗罗》一代的第一关、第五～第八关为横向卷动画面，第三关为纵向滚动画面；而《魂斗罗》二代则大多数关次采用了画面的横纵向混合位移技术；《上尉密令》游戏中则全部采用了横纵向混合位移技术。画面位移技术的应用使游戏画面广阔博大、连绵不绝，充分展现了背景的广度和深度，增强了游戏的真实感和参予感。关于整幅背景画面的位移控制处理方法、位移原理，大家已经十分熟悉，但对于游戏画面的分裂位移控制和画面的扭曲控制操作方法及控制原理则很陌生。如《大赛车》游戏中标题画面的三段位移结构，游戏主画面中的提示栏、远山、公路三段画面的分裂控制以及公路路面的扭曲控制则属于背景处理中的高难度技术。日本任天堂公司则把该项技术列入绝密级资料。因此，长期以来画面分裂控制技术不啻是一难解之谜。本章将通过《大赛车》游戏标题画面的绘制揭开这层神秘的面纱。

第一节　背景画面结构的新概念

一、背景画面结构复述

通过前几章的讨论，我们已经了解掌握了任天堂游戏背

景画面的结构和画面位移的基本原理，并明确了画面的各向位移机理。由此是否可以断定 PPU 对背景画面的处理就是：从屏幕显示映射单元读出图形代码，再根据选定的图形库，从代码对应的字模地址中读出字模数据来产生背景画面呢？回答是肯定的，但实际处理要复杂的多。

以往我们仅知道卡通可以以点为单位各向运动，背景画面可以以点为单位各向位移，背景画面和卡通均以 8×8 点阵为单位显示。实际上背景画面的显示也是以点为单位逐点处理的。为便于讨论，特对背景画面的结构复述如下。

我们知道背景画面的结构如图 8—1 所示。

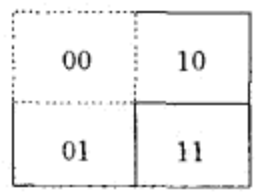


图 8—1　背景画面的结构

图 8—1 中双线框为屏幕显示窗口，00、01 页组成背景 0 页，10、11 页组成背景 1 页。

每一个显示页面由 960 个显示单元构成，每一个显示单元存储一个图形(图块)代码，每一个图形代码由图形库(字模区)中连续的 16 个数据定义，即 16 个数据说明 64 个点的状态，平均每一个数据定义图块中的 4 个点，即每两位二进制数显示一个点。

二、背景画面形成的操作过程

PPU 把每页背景画面以点为单位分成 256×240 点阵。在显示背景图形时，首先从显示页面的屏幕映射单元中读出图形代码；然后再根据选定的字模区读出该代码对应图形字

模的第一个数据和第九个数据，产生该图块第一行八个点的显示信号；随后再依次产生下一图形块第一行八个点的显示信号。当第一行的 256 个点全部处理完毕后，再处理另一显示页面的第一行的 256 个点。即 PPU 对背景画面的处理是以 512 个点为单位逐行处理的。当第一行处理完毕再处理第二行，直到处理完 240 行。

三、背景画面的两种描述

为叙述方便，我们把由 512 个点组成的一行称为一个点行(或线)。由此任天堂游戏画面将有两种描述：

一种描述是背景画面由四个显示页面组成，每个显示页面由 32 列×30 行共 960 个图块构成，屏幕显示窗口为 32 列×30 行，可将显示窗口移到由四个显示页面构成圆环的任意位置显示 32 列×30 行背景画面。

另一种描述为背景画面由四个显示页面 512 列×480 行个点构成，每个显示页面为 256 列×240 行点；每一点行长度为 512 点，每个显示页面由 240 个点行构成；背景显示窗口为 256×240 点阵，可将显示窗口移到画面的任意位置显示 256 列×240 行点阵画面。

显然，前一种描述属粗线条结构，对于画面的绘制讨论十分方便。后一种描述属细腻结构，便于画面的控制讨论。因此，两种描述相辅相成、互相补充，构成了任天堂游戏背景画面显示系统的完整理论。

四、背景画面位移的解释

需要特别指出的是：任天堂游戏机中只配备了一块 2K 字节的 VRAM，显然背景只有两个页面(00 页、10 页)，这是

根据任天堂游戏中背景显示的特点采取的一种廉价配置。

当游戏中需要画面横向位移时，则令两个显示页面横向排列如图 8－1 上半部分所示，对应两个页面的屏幕显示映射区为 $2000～$23BF、$2400～$27BF；当游戏中需要画面纵向位移时，则令两个显示页面纵向排列如图 8－1 左半部分所示(把 10 页放于 01 页的位置)，对应两个页面的屏幕显示映射区为 $2000～$23BF、$2700～$2BBF。这就是游戏编程中通常所说的横纵向控制(横纵向控制是通过游戏卡中的硬件电路和游戏软件共同作用控制的)。因此，当两个显示页面横向排列时，整个背景画面只有 512 列×240 行有效，下部的 240 行因无硬件支持不能显示；同样，当两个显示页面纵向排列时则只有 256 列×480 行有效。

第二节　背景画面的分裂位移

本节以《大赛车》标题画面结构为例介绍背景画面分裂位移的原理与控制方法。

一、背景画面形成点行的坐标

上一节曾谈到，PPU 对背景画面显示信号的产生是以点行为单位进行的。而每一点行在窗口的显示位置则仍由置入 $2005 单元的 X、Y 坐标控制。PPU 在产生每一个点行显示信号之前，首先检查点行首的显示坐标，这两个坐标参数在 PPU 内部由两个专用的寄存器存放。程序中只要对 $2005 单元连续置入两个数据，PPU 会自动按置入数据的先后顺序分别存入点行首显示坐标寄存器。PPU 处理点行的顺序是先 0 页后 1 页，即处理完背景 0 页的一个点行后，

接着处理背景 1 页的对应点行。因此，我们可以把一个点行理解为由背景 0 页、1 页两条线首尾相接组成的点环(或线环)，那么点环的接点即为点行显示坐标的参照点，置入 $2005 单元的 X 坐标值即为点行首距显示窗口左边沿的距离(向左超前)，其 Y 坐标值即为该点行距显示窗口上边沿的距离(向上超前)。但要注意，线环有两个接点，每一个接点既是前一页面点线的尾又是当前页面点线的首。如选定背景 0 页，对 $2005 单元置入 14H、00 时，若以点行首为坐标 0 点，显示窗口的对应关系见图 8－2。

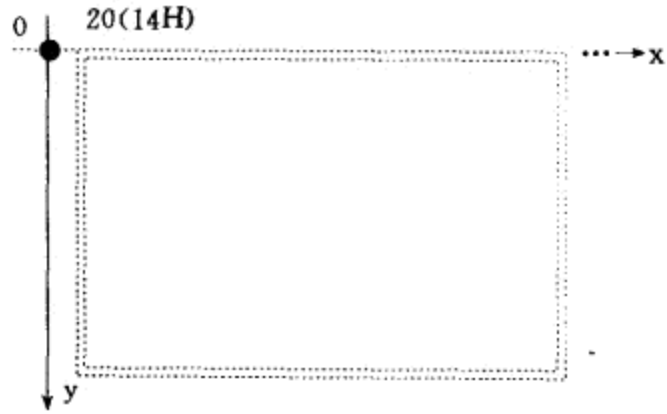


图 8－2　显示窗口与点行首坐标的对应关系图

当 PPU 处理完一个点行的视频信号后，再处理下一个点行，同样在处理这一点行之前，也要先从点行首坐标寄存单元读出 X、Y 坐标值，以决定该点行位于显示窗口的位置。若这时未对 $2005 单元置入新的坐标值，则仍以上一行的坐标为准。若这时对 $2005 单元置入了新的坐标值，则令当前处理的点行以新的坐标在显示窗口显示。试验测定 PPU 处理一个点行的时间约需 140 个机器周期。因此，只要控制置入 $2005 单元新坐标值的时间间隔，就可以控制画面位移的纵向宽度。

二、画面分裂位移程序

明确了点行首与显示窗口的对应关系后，就可以着手设计画面分裂位移的控制程序。《大赛车》标题画面的分裂位移程序如 No. 8－1。

No. 8－1 标题画面分裂位移程序

```
E72A A920     LDA # $20 ;
E72C A000     LDY # $00
E72E 2064E9   JSR $E964
E731 8C0520   STY $2005
E734 8C0520   STY $2005
E737 A991     LDA # $91
E739 8D0020   STA $2000
E73C A91E     LDA # $1E
E73E 8D0120   STA $2001
E741 A000     LDY # $00
E743 8417     STY $17
E745 88       DEY
E746 8418     STY $18 ;
E748 201BC9   JSR $C91B ;
E74B A517     LDA $17
E74D 8D0520   STA $2005
E750 A900     LDA # $00
E752 8D0520   STA $2005
E755 A8       TAY
E756 A991     LDA # $91
E758 8D0020   STA $2000
```

（一）画面分裂位移预备处理

置 PPU 地址 $2000，使画面的分裂位移从背景 0 页的第一行开始。置第一点行的显示坐标为(0,0)。

令显示窗口对正背景 1 页画面。

开显示。

对标题画面的上段位移控制变量置初值，令 $17＝0。

对标题画面的中段位移控制变量置初值，令 $18＝FF。

（二）背景 0 页位移控制处理

读中断。

令标题画面的由上段左向右移入显示窗口。

显示窗口对正背景 1 页。

延时中显示背景 1 页。

```
E75B A206     LDX # $06 ┐
E75D 2008FF   JSR $FF08 │
E760 CA       DEX       │
E761 10FA     BPL $E75D ;┘
```

```
E763 A518     LDA $18 ;  ┐
E765 8D0520   STA $2005  │
E768 8D0520   STA $2005  │
E76B A514     LDA $14    │
E76D 8D0020   STA $2000  │
E770 A204     LDX # $04  │
E772 2008FF   JSR $FF08  │
E775 CA       DEX        │
E776 10FA     BPL $E772 ;┘
```

(三)背景 1 页位移控制处理
令标题画面的中段由右向左移入显示窗口
$14 单元中的数据为 90。
令显示窗口对正背景 0 页。
延时中显示背景 0 页。

```
E778 A900     LDA # $00 ;┐
E77A 8D0520   STA $2005  │
E77D 8D0520   STA $2005 ;┘
```

(四)标题画面下段不动的控制处理。
下段不动时显示背景 0 页。

```
E780 A517     LDA $17   ;  ┐
E782 18       CLC          │
E783 6904     ADC # $04    │
E785 8517     STA $17      │
E787 A518     LDA $18      │
E789 38       SEC          │
E78A E904     SBC # $04    │
E78C 8518     STA $18      │
E78E A517     LDA $17      ┘
```

(五)画面位移控制指针的改变和位移结束的判断。
令标题画面的上段位移控制器加 4。
令标题画面的中段位移控制器减 4。
若 $17=FC 则结束位移。
若 $17<FC 则继续位移。

```
E790 C9FC     CMP # $FC
E792 F003     BEQ $E797
E794 4C48E7   JMP $E748 ;
E797 201BC9   JSR $C91B ;   (六)画面分裂位移结束后的画面定位处理
E79A A514     LDA $14       显示背景0页。
E79C 8D0020   STA $2000
E79F A900     LDA # $00     画面定位。
E7A1 8D0520   STA $2005
E7A4 8D0520   STA $2005 ;
```

三、画面分裂位移程序工作原理

从程序 No. 8－1 的注释可见，《大赛车》标题画面的分裂位移控制过程共经过六个阶段。第一、第五、第六阶段与常规程序相同，不多赘述。重点讨论第二、三、四阶段。

1. 标题画面上段的位移控制

首先读中断，只有等中断到来时才能进行位移操作。将 $17 单元中的点行首 X 坐标置入 $2005 单元，点行首 Y 坐标置 0。对 $2000 单元置入 91H 选背景 1 页显示，置 X＝6 控制连续调 $FF08 延时子程序 7 次，共延时 9081 个机器周期，接近处理完背景 1 页中 65 个点行的显示，从而控制标题画面上段先从 1 页的 X＝0 处开始由左向右移动，使“F－1”缓缓移入显示窗口。

2. 标题画面中段的位移控制

将 $18 单元的点行首 X 坐标置入 $2005 单元，Y 坐标置入 0，对 $2000 单元置入 90H(这时 $14＝90)选背景 0 页显示，置 X＝4 控制连续调 $FF08 延时子程序 5 次，共延时 6487 个机器周期，接近处理完背景 0 页中 47 个点行的显示，

从而控制标题画面中段从 0 页的 X＝FF 开始由右向左移动，使“RACE”缓缓移入显示窗口。

3. 标题画面下段的控制

置 $2005＝0、$2005＝0，即置以后点行的行首显示坐标为(0,0)，直到 PPU 处理完剩余点行的显示。从而使标题画面下段 128 点行所显示的游戏难度选项提示固定不动。

标题画面位移开始时，显示窗口中的画面与背景 0 页、1 页的排列位置见图 8－3。

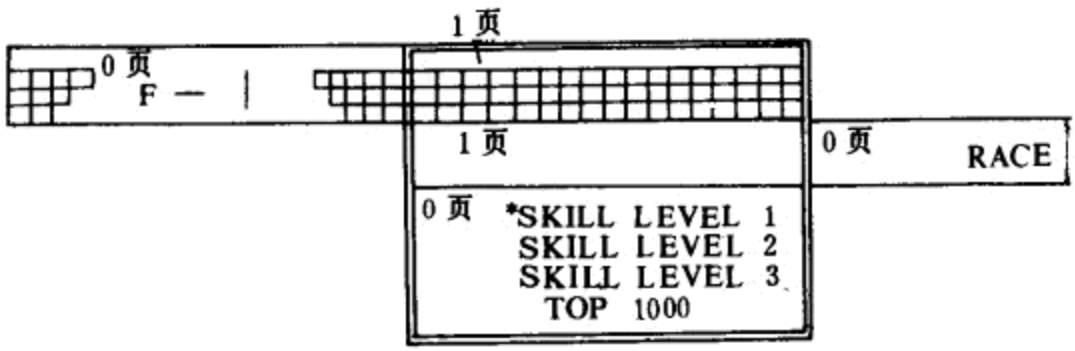


图 8－3　标题画面位移开始时的画面结构示意图、

当位移 32 次 $18＝80 时，画面刚好位移了 50%，这时的画面结构如图 8－4 所示。

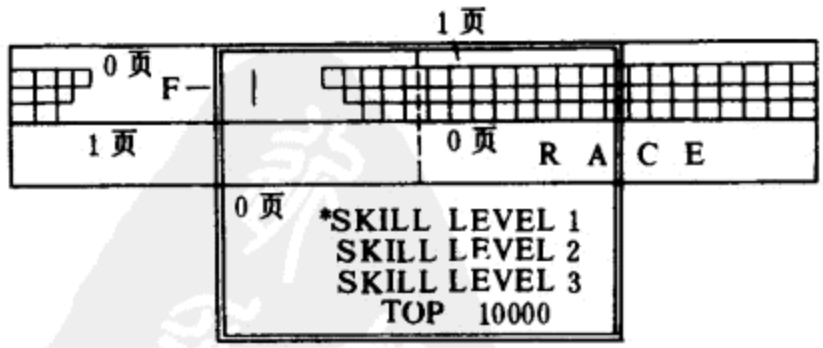


图 8－4　标题画面位移一半时的画面结构示意图

四、画面分裂位移控制方法

由以上讨论，可以总结出画面分裂位移的控制方法：

" 把一幅背景画面 32 列×30 行分为 240 个点行(或称为线)。

(位移开始时若选定哪一个页面对正显示窗口，可由程序 No. 8－2 控制。

No. 8－2 选分裂位移页面处理程序

```
LDA  # $20(或 24)
STA  $2006
LDY  # $00
STY  $2006
RTS
```

当选 0 页时，A 寄存器的值可取 20H～23H；

当选 1 页时，A 寄存器的值可取 24H～27H。

但 Y 寄存器的值必须为 0，若 Y≠0，整幅画面要向左上角位移与 Y 值对应的点数。

3. 通过对 $2000 单元的背景显示页面选择开关置值，选择欲显示内容的背景页面。

4. 对 $2005 单元置入当前欲显示内容位于显示窗口的 X、Y 坐标。

5. 通过控制延时时间，控制显示画面的纵向高度。

显然，通过对以上五个步骤的灵活组合，可以实现显示画面任意块分裂位移。

第三节　背景画面的扭曲控制

背景画面的扭曲技术在《侏罗记公园》游戏的标题画面中、《米老鼠》游戏的游戏接续画面中都有不同程度的使用，但使用最多的还是《大赛车》游戏。在该游戏主画面中，路面扭曲控制贯穿游戏始终。

我们在讨论了背景画面的分裂位移原理以后，已不难理解画面扭曲的控制原理和方法。实际上，画面扭曲控制是画面分裂位移的一种特殊控制形式。在画面的扭曲控制中，对每一个点行都赋予了新的显示坐标，使每一个点行的起始坐标相对于上一点行产生错位，从而产生了画面圆滑扭曲的效果。

可见，背景画面的分裂位移是从粗处着手，控制的是数个点行使用同一起点坐标组成的图块；背景画面扭曲则从细处着眼，控制每一个点行都以不同的起点坐标形成图形。

由于画面扭曲控制程序结构和控制处理过程都较复杂，必须结合具体的全部游戏程序清单和游戏进程进行具体分析，文字量是较大的，本丛书的篇幅已难以容纳详细的叙述。故只能简单说明画面扭曲的控制原理和方法，有兴趣的读者可参阅拙作《任天堂游戏编程探密》一书第六章的介绍。

总之，电脑学习机作图有着无穷的魅力和潜力，只要潜心揣摩、执著探索，定能展现奇风异彩，领略无穷的乐趣。同时，你的编程水平也将伴随着百尺竿头更进一步。

电脑学习机上机指导系列丛书

1. 操作入门
2. 文字处理
3. 游戏BASIC语言
4. 中文BASIC语言
5. LOGO语言
6. 汇编语言
7. 电脑作曲
8. 电脑绘图
9. BASIC语言程序集
10. 硬件与维修

ISBN 7-5053-3533-2

9 787505 335332 >

责任编辑：施玉新
封面设计：阎欢玲

ISBN7－5053－3533－2/TP·1428　定价：10.50 元